# Inheriting Generalized Learngene for Efficient Knowledge Transfer across Multiple Tasks

**Yuankun Zu**[1,2], **Shiyu Xia**[1,2], **Xu Yang**[1,2*], **Qiufeng Wang**[1,2], **Han Zhang**[3], **Xin Geng**[1,2*]

[1]School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
[2]Key Laboratory of New Generation Artificial Intelligence Technology and Its Interdisciplinary Applications (Southeast University), Ministry of Education, China
[3]School of Biological Science and Medical Engineering, Southeast University, Nanjing 210096, China
{zyk0418, shiyu_xia, xuyang_palm, qfwang, zhann, xgeng}@seu.edu.cn

## Abstract

In practical applications, it is often necessary to transfer knowledge from large pretrained models to small ones with various architectures for tackling different tasks. The *Learngene* framework, proposed recently, firstly extracts one compact module termed as **learngene** from a large well-trained model, after which learngene is used to build descendant models for handling diverse tasks. In this paper, we aim to explore extracting and inheriting learngene which can be generalized across different model architectures and tasks, remaining understudied in previous works. Inspired by the existing observations that large kernel convolutional neural networks (CNNs) exhibit significant generalization potential across various architectures and tasks, we propose a novel two-stage Learngene method termed CLKG (**C**onvolutional **L**earngene for **K**nowledge **G**eneralization), which inherits convolutional kernels containing generalized knowledge as learngene to build diverse models for multiple tasks. Specifically, we construct an auxiliary model comprised of small kernels and train it through dense feature distillation to inherit the feature extraction ability from large kernel CNNs. After distillation, we select certain kernels from the auxiliary model as learngene based on three criteria: direct kernel extraction, priority to edge kernels, and continuous kernel selection. Subsequently, we adapt learngene according to the width of the descendant models and use it to initialize the backbone part of descendant models. Experiments on diverse vision tasks such as image classification, object detection and semantic segmentation demonstrate the superiority of CLKG. For example, compared with from scratch training, it brings **2.89%** improvements on VOC12+SBD, and reduces around **2x** training data volume and training epochs to achieve better results. Furthermore, compared to knowledge distillation method, CLKG significantly reduces negative transfer on certain datasets, *e.g.* , achieves **1.88%** performance improvements on NAO dataset despite domain differences.

## Introduction

Deep neural networks demonstrate remarkable performance across a wide variety of computer vision tasks (Shu, Meng, and Xu 2023; Wichmann and Geirhos 2023; Yang et al. 2024; Zhao et al. 2024; Tang et al. 2024). In practical applications, it is essential to deploy and train models with different architectures for handling diverse tasks, where transferring well-learned knowledge from large pretrained models to compact target ones becomes increasingly important. Techniques such as knowledge distillation (KD) (Hinton, Vinyals, and Dean 2015) are frequently employed for this purpose, enhancing performance and training efficiency (Wu et al. 2023; Mistry 2024). Despite the success, it also faces several challenges, including the phenomenon of negative transfer when source and target tasks exhibit significant domain differences, which may harm the performance of the student model (Yang, Huang, and Wei 2023; Go et al. 2024). Furthermore, when we need student models with different architectures or sizes, repeated distillation processes are necessary as shown in Fig. 1(b).

To address this issue, a new framework *Learngene* was proposed (Wang et al. 2022; Xia et al. 2024a), which first extracts condensed knowledge termed as **learngene** from a well-trained ancestry model, and then inherit it into small or middle-sized descendant models to address downstream tasks, as shown in Fig. 1(c). The existing work Heur (Wang et al. 2022) extracts some integral layers as learngene according to gradient information during training. TLEG (Xia et al. 2023) proposes to linearly expand learngene layers to compose variable-depth descendant models. Learngene Pool (Shi et al. 2023) builds a pool of transformer layers by distillation and stiches them to create descendant models. SWS (Xia et al. 2024b) explores learngene in a multi-stage weight sharing way. Take a closer look at the Learngene framework, we note that descendant models frequently face various tasks, each also with varying architectures, which highlights the importance of **generalization** for learngene to enhance the adaptability and flexibility of descendant models across different tasks and architectures. However, existing methods extract learngene without considering the generalization across different tasks and architectures, greatly limiting its capability.

To this end, our study explores extracting and inheriting learngene which can be generalized to different model architectures for handling diverse vision tasks. Previous research suggests that large kernel convolutional neural networks (CNNs) exhibit significant generalization potential

*Co-corresponding author.

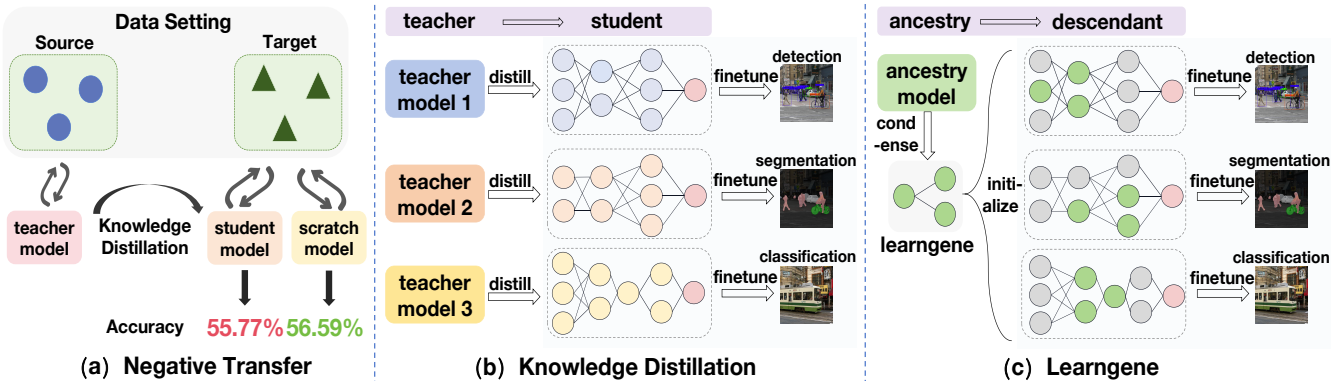


Figure 1: (a) Possible negative transfer (Wang et al. 2019; Zhang et al. 2022) phenomena in transfer learning methods such as KD when a mismatch between domains. (b) Knowledge distillation framework. (c) Learngene framework.

across various tasks, attributed to their expansive effective receptive field and pronounced shape bias, enabling them to benefit handling multiple tasks (Ding et al. 2022; Huang et al. 2023). Inspired by this, we propose a novel two-stage Learngene method called CLKG (**C**onvolutional **L**earngene for **K**nowledge **G**eneralization). In the first stage, we term the feature extraction ability of certain convolution kernels as generalized knowledge and extract it as learngene. In the second stage, we inherit learngene to initialize descendant models with different architectures for diverse vision tasks.

Specifically, in the first stage, to extract learngene, we construct an auxiliary model with small kernels to approximate the feature extraction ability of large kernel CNNs by dense feature distillation to align the multi-stage feature maps (Wei et al. 2022). After distillation, certain convolutional kernels in the auxiliary model retain robust feature extraction abilities (Yamashita et al. 2018). In this way, we select the kernels as learngene based on a combination of three criteria: direct kernel extraction, priority to edge kernels, and continuous kernel selection.

After obtaining the learngene, in the second stage, we adapt them to descendant models with different architectures. For wider descendant models that require more kernels, we introduce additional small weight kernels that conform to Gaussian distribution (Kumar 2017). For narrower models which requires fewer kernels, we prioritize kernels based on their L2 norm and select those with the highest values as they likely contain more information (He et al. 2015; Zhang et al. 2023). Specifically when adjusting their width, we modify the number of convolutional kernels in one layer. In addition, we use 1x1 convolutions (Szegedy et al. 2015) to adjust the number of channels when dealing with descendant models that have special structures. After these adjustments, the learngene can directly initialize specific convolutional layers in the backbone part of the descendant model.

Compared with training from scratch, CLKG has a significant improvement in performance, *e.g.* , **+3.64%** on mini-ImageNet (Vinyals et al. 2016), and reduces around **2x** training data volume or training epochs to achieve better results. Moreover, learngene extracted by CLKG can be generalized to different model architectures across diverse tasks effectively, *e.g.* , **+2.59%** on COCO2017 and **+2.89%** on VOC12+SBD. In addition, compared with KD (Zagoruyko and Komodakis 2016; Wei et al. 2022) methods, CLKG not only significantly reduce computational resources by distilling at most once, but also greatly reduces negative transfer through inheriting generalized knowledge, *e.g.* , achieves **2.83%** performance gains on Oxford_Flower (Nilsback and Zisserman 2008), **1.88%** performance gains on NAO (Lau et al. 2021), despite domain differences. Our main **contributions** are summarized as follows:

- We are the first to explore the generalization potential of learngene across different tasks and architectures by extracting and inheriting feature extraction ability from large kernel CNN models.
- We propose a novel two-stage Learngene method termed CLKG, which includes the design of learngene selection and adaptation for diverse descendant models.
- Extensive experiments demonstrate the key benefits of CLKG, such as training data efficiency and the flexibility of model architectures. Besides, we also explore the phenomenon of negative transfer.

## Related Work

### Learngene

Learngene is a two-stage framework which firstly learns one compact module termed as **learngene** from a large well-trained network called ancestry model, and then inherit it to initialize descendant models as shown in Fig. 1(c). Heur (Wang et al. 2022) extracts entire layers based on gradient information as learngene, combining them with randomly initialized ones to form descendant models. TLEG (Xia et al. 2023) linearly expands two transformer learngene layers, trains them via distillation, and uses them to initialize transformers of various depths. Learngene Pool (Shi et al. 2023) distills a pretrained ancestry model into multiple small ones as learngene and then stitches them to build descendant models. SWS (Xia et al. 2024b) extracts learngene in a multi-stage weight sharing fashion, where learngene retains stage-specific information for initializing descendant models. Revisiting the Learning framework, we

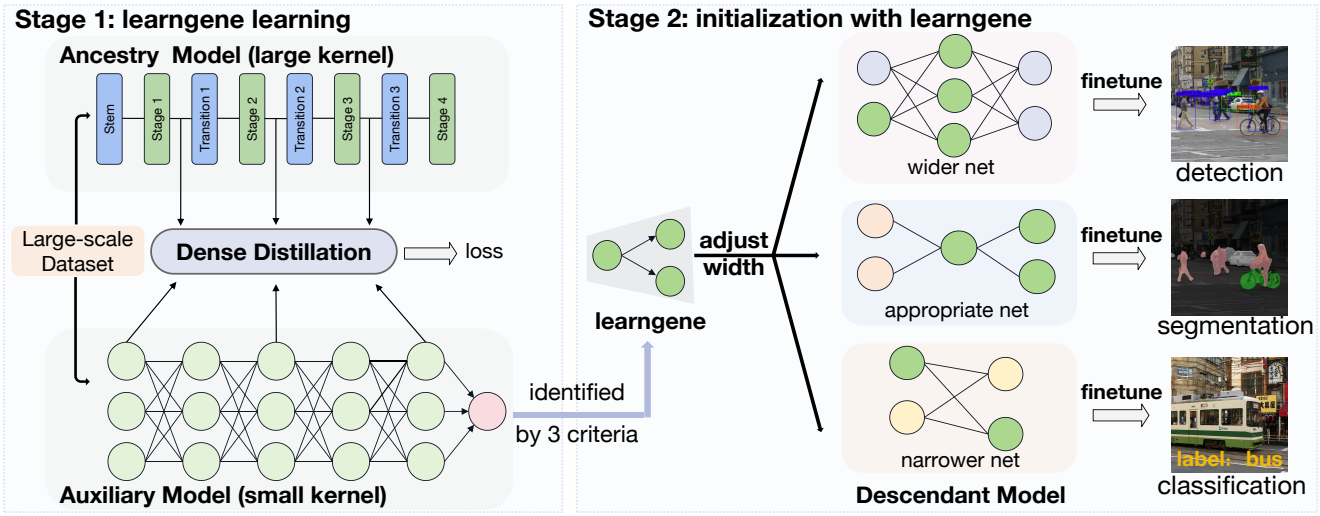


Figure 2: The technical details of CLKG. In the first stage, we design a CNN consisting of small kernels as an auxiliary model and use dense feature distillation to extract knowledge from the large kernel ancestry model. Then, based on three criteria, we select some convolutional kernels as learngene. In the second stage, we adjust the learngene width to enable them to initialize descendant models of different architectures, and finetune them on multiple downstream tasks.

find that descendant models frequently face diverse and new tasks, accompanied by different architectures, which highlights the importance of **generalization** for learngene to enhance the adaptability of descendant models across different tasks and architectures. However, existing methods often overlook the necessity for generalization, which may restrict its capability to adapt to broader tasks.

### Knowledge Distillation

Knowledge distillation (KD) (Hinton, Vinyals, and Dean 2015; Zagoruyko and Komodakis 2016; Gou et al. 2021; Wang and Yoon 2021) aims at transferring knowledge from a large, complex model (teacher model) to a smaller, more efficient model (student model) to improve the performance of the student model while reducing computational and storage costs. However, these KD methods require re-distillation for different student architectures, significantly increasing computational demands. Additionally, transferring all acquired knowledge may lead to negative transfer, where using source domain knowledge reduces learning performance in the target domain (Yang, Huang, and Wei 2023; Go et al. 2024). Solutions like using multiple teachers (Ji et al. 2023) or advanced distillation techniques (Jacob, Agarwal, and Stenger 2023; Ding, Yang, and Zheng 2024) attempt to address this, where they further complicate the training process and increase resource consumption. In contrast, our method selectively transfers generalized knowledge, ensuring a more flexible and effective adaptation to different tasks.

### Models with Large Kernels

In contrast to small kernel CNNs, large kernel CNNs have larger effective receptive fields and higher shape bias (Luo et al. 2016). Benefiting from this, when using a large-kernel CNN model as the backbone, it demonstrates strong generalization, enabling it to exhibit superior performance across multiple tasks and architectures (Ding et al. 2022; Huang et al. 2023). Due to the fact that the backbone of a model is typically used for initial feature extraction (Xue et al. 2023), in this paper, we attempt to inherit the feature extraction capabilities of large kernel CNNs to enhance generalization.

## Method

### Overview

Fig. 2 depicts the overall pipeline of CLKG. In Stage 1, we construct an auxiliary model composed of convolutional layers with 3x3 kernels of varying numbers. Subsequently, we employ dense feature distillation (Heo et al. 2019) to approximate the feature extraction capability of large kernel models. Following this, certain convolutional kernels of the auxiliary model are identified as learngene based on three criteria. In Stage 2, according to the architecture of descendant models, we make adjustments on the width of learngene. Then, we can use learngene to directly initialize some convolutional kernels in the backbone part of the descendant model. Finally, we finetune the descendant models conventionally on specific tasks.

### Learngene Extraction Process

**Training the auxiliary model** In our study, we choose RepLKNet (Ding et al. 2022) as the ancestry model for its significant generalization potential across various tasks. Its clear multi-stage structure is ideal for guiding small CNN student models through feature distillation.

In order to extract generalized knowledge from the ancestry model, we design an auxiliary model to approximate the feature extraction ability of the large kernel model. With the theory that stacking two 3x3 convolution layers achieves the same receptive field size as a single 5x5 convolution

layer and adding pooling layers can obtain a larger receptive field (Simonyan and Zisserman 2014; Luo et al. 2016), we design our auxiliary model to satisfy a hierarchical process consisting of small kernels. Specifically, our auxiliary model is divided into three stages:

$$F_x^3 = f_x^1 \circ f_x^2 \circ f_x^3, \tag{1}$$

where $f_x^i$ represents the i-th stage of the auxiliary model.

To distill knowledge from the ancestry model for various tasks, we introduce dense feature distillation to approximate the output of the first three stages of the large kernel model to calculate the distillation loss function, capturing hierarchical representations (Zhang, Bengio, and Singer 2022). Specifically, we distill information from the final layer of each stage in the ancestry model to the corresponding layer in the auxiliary model. Thus the loss function is denoted as:

$$\mathcal{L}_{FD} = \sum_{i=1}^{3} \alpha_i \| g(T^i) - w(S^i) \|_2, \tag{2}$$

where $T^i, S^i$ denotes the last layer of the i-th stage feature map of the teacher and the student, respectively. $g$ is a 1×1 convolution layer to align the number of channels. $w$ is a non-parametric layer normalization for the whitening operation (Yong and Zhang 2022). $\alpha$ denotes the trade-off.

**Three Criteria for Identifying Learngene** After distillation, specific convolutional kernels of the auxiliary model retain feature extraction abilities (Yamashita et al. 2018), signifying their possession of generalized knowledge. In this way, these kernels are identified as learngene based on a combination of the following three criteria: direct kernel extraction, priority to edge kernels, and continuous kernel selection. For a detailed explanation for the selection from the auxiliary model rather than from the ancestry model, please see Section A.2 of the Appendix.

1) **Direct kernel extraction.** In our method, we extract convolutional kernels directly from the layers of the auxiliary model. Based on the receptive field theory (Luo et al. 2016), the effective receptive field is proportional to $K\sqrt{L}$ (where $K$ is the kernel size and $L$ is the depth). Extracting features directly from images using small kernel models alone often yields suboptimal results (Ding et al. 2022; Huang et al. 2023). Therefore, we employ feature map alignment, allowing a large kernel model to supervise the training of the small kernel model. This approach ensures that some convolutional kernels in the auxiliary model can effectively learn the feature extraction capabilities of the large kernel ancestry model, allowing us to directly extract small kernels from the auxiliary model.

2) **Priority to edge kernels.** During the model training process, we find that the weight changes in the middle layer of the model are very small. A recent study on ViTs and large language models corroborated this finding (Samragh et al. 2023), which also found that deleting or replicating weights in intermediate layers has minimal impact on performance. Therefore, in selecting learngene, we focus on edge kernels, *i.e.* , convolutional kernels from both the initial and final stages of the network. Besides, we do not completely ignore the contribution of the intermediate stage. Instead, we use feature loss (Eq. 2) as a soft constraint to reflect the contribution of the intermediate stage in the weights of the first and last stages of the network.

3) **Continuous kernel selection**. Features in CNNs are learned in a hierarchical manner, with convolutional kernels in each layer capturing increasingly abstract features (Yosinski et al. 2015; Zhang, Bengio, and Singer 2022). Inheriting convolutional kernels from a single layer disrupts this hierarchy, potentially causing a loss of essential feature information. By selecting continuous convolutional kernels across layers, we ensure the preservation of feature hierarchy and continuity, allowing for meaningful and effective knowledge transfer.

## Inheriting Learngene into Descendant Model

In CNNs, targeted architectures differ in the latter half of the network for tasks such as image classification, object detection, and semantic segmentation, with each task focusing on different aspects of the image. However, the initial feature extraction stages often share similarities. Therefore, we only use learngene to initialize the backbone part of the descendant models. We denote the convolutional layer as $W_i$, where $n_i$ represents the number of kernels, $c_i$ the channels, and $k_i$ the kernel size. Consequently, $W_i[n_i; c_i; k_i; k_i]$ describes a layer with $n_i$ kernels of size $k_i \times k_i$ and $c_i$ channels, with $i = 1$ referring to learngene layers and $i = 2$ to those in descendant models. When the architecture of the descendant model closely aligns with that of the extracted learngene, we replace some randomly initialized convolutional kernels with learngene. However, mismatches often occur in the width of kernels, particularly in the number of convolutional kernels and channels, thus we need to adapt learngene for initialize these parts.

**Learngene Adaptation** To address discrepancies in kernel numbers, particularly when $n_1 < n_2$, we transfer the weights of $k_1$ convolutional kernels from the learngene. For the remaining convolutional kernels, we initialize them with small random numbers drawn from a Gaussian distribution (Kumar 2017). This process is represented as follows:

$$W_2[n_2; c; k; k] = W_1[n_1; c; k; k] + W_{\text{new}}, \tag{3}$$

where $W_{\text{new}} \sim \mathcal{N}(0, 0.05^2)$ and $W_{\text{new}} \in \mathcal{R}^{(n_2-n_1)\times c\times k\times k}$.

When $n_1 > n_2$, we believe that convolutional kernels with larger norm play a more important role in convolutional layers (He et al. 2015; Zhang et al. 2023). Our definition of convolutional kernel norm is as follows:

$$\|L_i\|_2 = \sqrt{\sum_{j,m,n} (w[i; j; m; n])^2}\,, i = 1, \ldots, n_1. \tag{4}$$

Using this norm, we sort the convolutional kernels and select the top $n_2$ with the highest norms to initialize the descendant network. This selection ensures that the most influential features are retained in the new model configuration.

In addition, we use a 1x1 convolution kernel to adjust the number of channels when dealing with descendant models that have special structures, making it difficult to modify the number of convolution kernels directly.

| task | model | dataset | Scratch | Heur (Wang et al. 2022) | CLKG |
|---|---|---|---|---|---|
| image classification | VGG16 | miniImageNet | 75.70 | 73.35 | **79.34 (+3.64)** |
| | | CIFAR-100 | 74.06 | 72.76 | **76.69 (+2.63)** |
| | | Oxford_Flower | 86.19 | 84.76 | **89.02 (+2.83)** |
| | BotNet | miniImageNet | 82.93 | 81.74 | **85.90 (+2.97)** |
| | | CIFAR-100 | 75.91 | 74.83 | **77.24 (+1.33)** |
| | | Oxford_Flower | 88.53 | 87.73 | **89.94 (+1.41)** |
| | ResNet50 | miniImageNet | 82.41 | 81.96 | **84.64 (+2.23)** |
| | | CIFAR-100 | 75.03 | 74.38 | **76.74 (+1.71)** |
| | | Oxford_Flower | 87.64 | 85.47 | **89.78 (+2.14)** |
| object detection | YOLOV7 | VOC07+12 | 80.16 | 78.23 | **82.81 (+2.65)** |
| | | COCO2017 | 64.23 | 61.42 | **66.82 (+2.59)** |
| | | NAO | 15.22 | 12.72 | **17.10 (+1.88)** |
| | DETR | VOC07+12 | 76.06 | 75.84 | **78.42 (+2.36)** |
| | | COCO2017 | 60.32 | 58.43 | **62.24 (+1.92)** |
| | | NAO | 15.34 | 13.68 | **16.87 (+1.53)** |
| semantic segmentation | DeepLabv3p | VOC12+SBD | 54.96 | 52.16 | **57.85 (+2.89)** |
| | | mini_ADE20K | 56.59 | 53.47 | **58.12 (+1.53)** |
| | Mobile-ViT | VOC12+SBD | 58.48 | 57.36 | **60.59 (+2.11)** |
| | | mini_ADE20K | 59.42 | 57.68 | **62.37 (+2.95)** |

Table 1: Performance comparison across different datasets on various popular CNN architectures among Scratch, Heur (Wang et al. 2022), and CLKG.

# Experiments

## Experimental Setup

**Datasets and Architectures** To learn learngene, we train the auxiliary model on ImageNet-1K (Deng et al. 2009). After initializing the descendant models with learngene, we further finetune them on several downstream datasets including miniImageNet (Vinyals et al. 2016), CIFAR-100 (Krizhevsky and Hinton 2009), Oxford-Flower (Nilsback and Zisserman 2008), VOC07+12 (Everingham et al. 2007), COCO2017 (Lin et al. 2014), NAO (Lau et al. 2021), VOC12+SBD (Hariharan et al. 2011) and mini-ADE20K (Zhou et al. 2019), covering three vision tasks: image classification, object detection, and semantic segmentation. We select RepLKNet-XL (Ding et al. 2022) as the ancestry model. For descendant model architectures, we select VGG16 (Simonyan and Zisserman 2014) and ResNet50 (He et al. 2016) for image classification tasks, YOLOV7 (Wang, Bochkovskiy, and Liao 2023) for object detection tasks, and DeepLabv3p (Chen et al. 2018) for semantic segmentation tasks. Moreover, we explore the CNN-Transformer hybrid models (Zhu et al. 2020; Srinivas et al. 2021; Mehta and Rastegari 2021). Given that CLKG is based on convolutional kernels of convolutional neural networks, this paper will not involve pure Vision Transformer architecture.

**Comparison Method** Several important baselines are compared: (1) Scratch: randomly initializes descendant models. (2) Heur (Wang et al. 2022): initializes descendant models with the last three layers of the ancestry model. (3) FT (Wei et al. 2022): performs feature distillation on descendant models from the same ancestry model for a fair comparison. (4) OKD-MTL (Jacob, Agarwal, and Stenger 2023): uses adaptive feature distillation loss and online task weighting to avoid negative transfer. (5) DASFD (Ding, Yang, and Zheng 2024): selectively learns positive features of objects by integrating depth uncertainty into feature distillation to avoid negative transfer. We do not compare with the TLEG and SWS methods because they both select the complete layers in the transformer for learngene, while we only need some convolutional kernels.

## Key Benefits of CLKG

We conduct experiments to verify four key benefits of CLKG: (i) Better Performance. (ii) Less Training Data Volume. (iii) Faster Convergence. (iv) Different architectures, which is **distinct** from previous Learngene methods.

**Better Performance** As shown in Table 1, CLKG demonstrates better performance across multiple diverse datasets. Specifically, compared with Scratch and Heur, it brings more than **2%** performance improvement, *e.g.* , **+2.63%** on CIFAR-100. Illustrated in Fig. 3 (a)-(c), CLKG consistently outperforms the model trained from scratch, while also reducing training epochs by approximately **2x**. This confirms its ability to transfer knowledge from the large kernel model to the small kernel model, thereby demonstrating the utility of inherited knowledge.

**Less Training Data Volume** As shown in Fig. 3 (d)-(f), CLKG reduces the training data volume randomly by around **2x** to achieve comparable results to the model trained from scratch, demonstrating its efficiency in enabling descendant models to flexibly adapt to different tasks. Additionally, when the training data volume is reduced, the performance of models trained from scratch significantly decreases, whereas CLKG maintains its performance.

**Faster Convergence** As shown in Fig. 3 (g)-(i), we find that CLKG outperforms the model trained from scratch

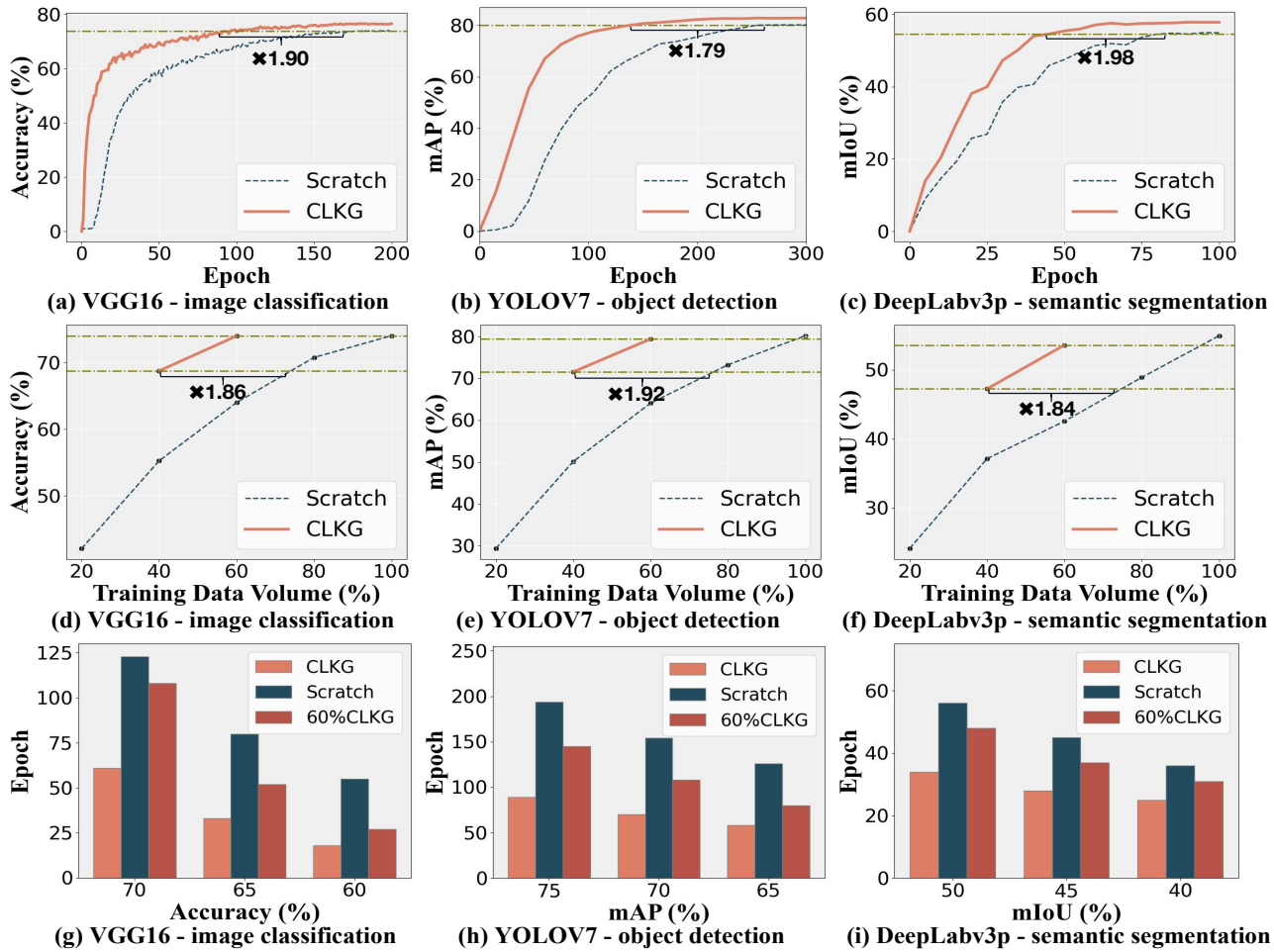


Figure 3: Comparison with learning from scratch on downstream tasks from left to right. (a)-(c) are about training performance. (d)-(f) are about data volume. (g)-(i) are about convergence speed. 60%CLKG means CLKG with 60% training data.

in terms of achieving promising performance within fewer training epochs, requiring around **half** of the training epochs. Remarkably, this superiority is maintained even when trained on merely **60%** data, which indicates that CLKG is effective in providing a well-initialized starting point for the descendant model.

**Different Architectures** As shown in Table 1, within many popular CNN models of different architectures, CLKG outperforms Scratch by a large margin, *e.g.* , **+2.59%** on COCO2017 dataset and **+2.89%** on VOC12+SBD dataset, demonstrating its effectiveness under settings of different model architectures. This benefit is **unique** to CLKG. Previous Learngene methods only emphasize models of different sizes but overlooks exploring models with different architectures, which is important in practical applications. As shown in Table 1, compared with Heur (Wang et al. 2022), the performance improvement is around **5%**, *e.g.* , **+5.40%** on COCO2017 dataset and **+5.69%** on VOC12+SBD dataset. This demonstrates the invalidity of Heur when facing architectural differences, which replaces the last several layers of a descendant model with those of the ancestry model.

## Comparison with KD on Negative Transfer

Due to the fact that knowledge distillation (KD) conveys all knowledge without distinguishment, it may contain knowledge or details that are not completely related to the downstream task, and may even result in poorer performance than Scratch. In practical tasks, we often face fine-grained data, unprocessed data in natural scenes, and data that does not match due to merging categories according to certain specific requirements (Nilsback and Zisserman 2008; Zhou et al. 2019; Lau et al. 2021) . As shown in Table 2, we find negative transfer phenomenon caused by KD in three different downstream tasks, while CLKG, due to containing generalized knowledge, reduces uncertainty in the learning process and has more flexible learning ability, thereby achieving better performance. Specifically, compared to Scratch, CLKG brings a **2.83%** performance improvement on Oxford_Flower, while FT(Wei et al. 2022) brings a **0.81%** performance decrease. Moreover, we compare CLKG with the latest method that avoids negative transfer through enhancing distillation process in a single teacher model. And the result shows that CLKG achieves better performance.

| | image classification<br>VGG16<br>Oxford_Flower | object detection<br>YOLOV7<br>NAO | semantic segmentation<br>DeepLabv3p<br>mini_ADE20K |
|---|---|---|---|
| Scratch | 86.19 | 15.22 | 56.59 |
| FT (Wei et al. 2022) | 85.38 | 14.54 | 55.77 |
| OKD-MTL (Jacob, Agarwal, and Stenger 2023) | 88.47 | 15.63 | 57.37 |
| DASFD (Ding, Yang, and Zheng 2024) | 87.34 | 15.74 | 57.13 |
| CLKG | **89.02 (+0.55)** | **17.10 (+1.36)** | **58.12 (+0.75)** |

Table 2: Comparison against KD methods on different tasks to explore the negative transfer phenomenon.

| model | CLKG | | |
|---|---|---|---|
| | wd | wid | d |
| VGG16 | 74.67 | 75.86 | **76.69 (+0.83)** |
| YOLOV7 | 81.08 | 82.27 | **82.81 (+0.54)** |
| DeepLabv3p | 56.18 | 56.83 | **57.85 (+1.02)** |

Table 3: Performance comparison among different KD strategies on CLKG. wd: only aligns the output of last stage of the ancestry model, wid: aligns the outputs of first stage and last stage of the ancestry model, d: aligns the outputs of all three stages of the ancestry model.

| model | Scratch | ancestry model | |
|---|---|---|---|
| | | Swin-T | RepLKNet-XL |
| VGG16 | 74.06 | 75.83 | **76.69 (+0.86)** |
| YOLOV7 | 80.16 | 82.44 | **82.81 (+0.37)** |
| DeepLabv3p | 54.96 | 57.25 | **57.85 (+0.60)** |

Table 4: Performance comparison on descendant CNN models across various tasks to evaluate the guidance competence between transformer and large kernel CNN ancestry model.

## Ablation Studies

In this section, we discuss different feature distillation methods and the selection of the ancestry model architecture. The datasets used for the ablation experiments are CIFAR-100, VOC07+12, and VOC12+SBD.

**Feature Distillation Method** We compare the feature distillation methods of selecting feature layers at different stages to compute loss function as shown in Table 3. We can see that the wd method is not ideal due to ignoring the output effects of other large kernels, while wid indicates that the output of the intermediate layer in network training should also be fully considered. The above results indicate the effectiveness of the dense feature distillation we employ.

**Selection of the Ancestry Model Architecture** In order to verify the generalization ability of the learngene extracted from the large kernel CNN, we select Swin-T (Liu et al. 2021), which also has a large receptive field, for comparsion. The KD method from Swin-T to small kernel auxiliary model refers to (Huang et al. 2023) to ensure fair comparison. As shown in Table 4, we observe that the descendant models guided by large kernel CNN RepLKNet-XL outperforms those guided by Swin-T, indicating the superiority of large kernel CNN models in guiding small kernel CNN models due to the similarity of architecture.

## Visualization Results

In this section, we visually compare feature maps from models initialized with CLKG and from scratch to evaluate the influence of the knowledge condensed by CLKG from a large kernel model on the downstream task datasets. The visualizations, based on grad-cam (Selvaraju et al. 2017), are shown in Fig. 4. These demonstrate that CLKG-initialized models focus more on salient image information, which is beneficial for various downstream tasks, thus providing a more effective initialization for subsequent learning process.

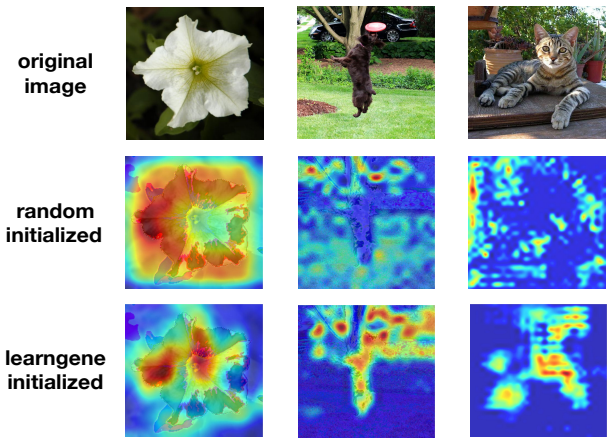


Figure 4: Visualization results of the feature maps of the last convolutional layer in the randomly initialized models and models initialized by CLKG.

## Conclusion

In this paper, we introduce a novel method for knowledge transfer in neural networks named CLKG. We approximate the feature extraction ability of the large kernel CNN model using a small kernel CNN model, and inherit the extracted convolutional kernels to descendant models of different architectures to adapt to different tasks. Extensive experiments validate our effectiveness and efficiency. In addition, we also verify that CLKG not only has advantages in computational resources and cost compared to the KD method, but can also avoid some negative transfer phenomena.

## Acknowledgments

This research is supported by Key Program of Jiangsu Science Foundation (BK20243012), the National Science Foundation of China (62125602, 62076063), the Fundamental Research Funds for the Central Universities(2242024k30035) and the Big Data Computing Center of Southeast University.

## References

Chen, L.-C.; Zhu, Y.; Papandreou, G.; Schroff, F.; and Adam, H. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, 801–818.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Ding, R.; Yang, M.; and Zheng, N. 2024. Selective Transfer Learning of Cross-Modality Distillation for Monocular 3D Object Detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 1–1.

Ding, X.; Zhang, X.; Han, J.; and Ding, G. 2022. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11963–11975.

Everingham, M.; Van Gool, L.; Williams, C. K. I.; Winn, J.; and Zisserman, A. 2007. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

Go, H.; Lee, Y.; Lee, S.; Oh, S.; Moon, H.; and Choi, S. 2024. Addressing negative transfer in diffusion models. *Advances in Neural Information Processing Systems*, 36.

Gou, J.; Yu, B.; Maybank, S. J.; and Tao, D. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6): 1789–1819.

Hariharan, B.; Arbeláez, P.; Bourdev, L.; Maji, S.; and Malik, J. 2011. Semantic contours from inverse detectors. In *2011 international conference on computer vision*, 991–998. IEEE.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Heo, B.; Kim, J.; Yun, S.; Park, H.; Kwak, N.; and Choi, J. Y. 2019. A comprehensive overhaul of feature distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1921–1930.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Huang, T.; Yin, L.; Zhang, Z.; Shen, L.; Fang, M.; Pechenizkiy, M.; Wang, Z.; and Liu, S. 2023. Are large kernels better teachers than transformers for convnets? In *International Conference on Machine Learning*, 14023–14038. PMLR.

Jacob, G. M.; Agarwal, V.; and Stenger, B. 2023. Online Knowledge Distillation for Multi-Task Learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2359–2368.

Ji, Z.; Ni, J.; Liu, X.; and Pang, Y. 2023. Teachers cooperation: team-knowledge distillation for multiple cross-domain few-shot learning. *Frontiers of Computer Science*, 17(2): 172312.

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*.

Kumar, S. K. 2017. On weight initialization in deep neural networks. *arXiv preprint arXiv:1704.08863*.

Lau, F.; Subramani, N.; Harrison, S.; Kim, A.; Branson, E.; and Liu, R. 2021. Natural adversarial objects. *arXiv preprint arXiv:2111.04204*.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, 740–755. Springer.

Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; and Guo, B. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, 10012–10022.

Luo, W.; Li, Y.; Urtasun, R.; and Zemel, R. 2016. Understanding the effective receptive field in deep convolutional neural networks. *Advances in neural information processing systems*, 29.

Mehta, S.; and Rastegari, M. 2021. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*.

Mistry, J. 2024. Automated Knowledge Transfer for Medical Image Segmentation Using Deep Learning. *Journal of Xidian University*, 18(1): 601–610.

Nilsback, M.-E.; and Zisserman, A. 2008. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, 722–729. IEEE.

Samragh, M.; Farajtabar, M.; Mehta, S.; Vemulapalli, R.; Faghri, F.; Naik, D.; Tuzel, O.; and Rastegari, M. 2023. Weight subcloning: direct initialization of transformers using larger pretrained ones. *arXiv preprint arXiv:2312.09299*.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, 618–626.

Shi, B.; Xia, S.; Yang, X.; Chen, H.; Kou, Z.; and Geng, X. 2023. Building Variable-sized Models via Learngene Pool. *arXiv preprint arXiv:2312.05743*.

Shu, J.; Meng, D.; and Xu, Z. 2023. Learning an explicit hyper-parameter prediction function conditioned on tasks. *The Journal of Machine Learning Research*, 24(1): 8818–8891.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Srinivas, A.; Lin, T.-Y.; Parmar, N.; Shlens, J.; Abbeel, P.; and Vaswani, A. 2021. Bottleneck transformers for visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 16519–16529.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Tang, W.; Yang, Y.-F.; Wang, Z.; Zhang, W.; and Zhang, M.-L. 2024. Multi-Instance Partial-Label Learning with Margin Adjustment. In *Advances in Neural Information Processing Systems 37, Vancouver, Canada*.

Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D.; et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems*, 29.

Wang, C.-Y.; Bochkovskiy, A.; and Liao, H.-Y. M. 2023. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 7464–7475.

Wang, L.; and Yoon, K.-J. 2021. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE transactions on pattern analysis and machine intelligence*, 44(6): 3048–3068.

Wang, Q.-F.; Geng, X.; Lin, S.-X.; Xia, S.-Y.; Qi, L.; and Xu, N. 2022. Learngene: From open-world to your learning task. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 8557–8565.

Wang, Z.; Dai, Z.; Póczos, B.; and Carbonell, J. 2019. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11293–11302.

Wei, Y.; Hu, H.; Xie, Z.; Zhang, Z.; Cao, Y.; Bao, J.; Chen, D.; and Guo, B. 2022. Contrastive learning rivals masked image modeling in fine-tuning via feature distillation. *arXiv preprint arXiv:2205.14141*.

Wichmann, F. A.; and Geirhos, R. 2023. Are deep neural networks adequate behavioral models of human visual perception? *Annual Review of Vision Science*, 9(1): 501–524.

Wu, Z.; Sun, S.; Wang, Y.; Liu, M.; Jiang, X.; Li, R.; and Gao, B. 2023. Survey of knowledge distillation in federated edge learning. *arXiv preprint arXiv:2301.05849*.

Xia, S.; Zhang, M.; Yang, X.; Chen, R.; Chen, H.; and Geng, X. 2023. Transformer as Linear Expansion of Learngene. *arXiv preprint arXiv:2312.05614*.

Xia, S.; Zu, Y.; Yang, X.; and Geng, X. 2024a. Initializing Variable-sized Vision Transformers from Learngene with Learnable Transformation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Xia, S.-Y.; Zhu, W.; Yang, X.; and Geng, X. 2024b. Exploring Learngene via Stage-wise Weight Sharing for Initializing Variable-sized Models. *arXiv preprint arXiv:2404.16897*.

Xue, G.; Li, S.; Hou, P.; Gao, S.; and Tan, R. 2023. Research on lightweight Yolo coal gangue detection algorithm based on resnet18 backbone feature network. *Internet of Things*, 22: 100762.

Yamashita, R.; Nishio, M.; Do, R. K. G.; and Togashi, K. 2018. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9: 611–629.

Yang, Y.; Guo, J.; Li, G.; Li, L.; Li, W.; and Yang, J. 2024. Alignment efficient image-sentence retrieval considering transferable cross-modal representation learning. *Frontiers Comput. Sci.*, 18(3): 181335.

Yang, Y.; Huang, L.-K.; and Wei, Y. 2023. Concept-wise Fine-tuning Matters in Preventing Negative Transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 18753–18763.

Yong, H.; and Zhang, L. 2022. An embedded feature whitening approach to deep neural network optimization. In *European Conference on Computer Vision*, 334–351. Springer.

Yosinski, J.; Clune, J.; Nguyen, A.; Fuchs, T.; and Lipson, H. 2015. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*.

Zagoruyko, S.; and Komodakis, N. 2016. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*.

Zhang, C.; Bengio, S.; and Singer, Y. 2022. Are all layers created equal? *Journal of Machine Learning Research*, 23(67): 1–28.

Zhang, L.; Wang, R.; Li, Z.; Li, J.; Ge, Y.; Wa, S.; Huang, S.; and Lv, C. 2023. Time-series neural network: a high-accuracy time-series forecasting method based on Kernel filter and time attention. *Information*, 14(9): 500.

Zhang, W.; Deng, L.; Zhang, L.; and Wu, D. 2022. A survey on negative transfer. *IEEE/CAA Journal of Automatica Sinica*, 10(2): 305–329.

Zhao, X.; Wang, L.; Zhang, Y.; Han, X.; Deveci, M.; and Parmar, M. 2024. A review of convolutional neural networks in computer vision. *Artificial Intelligence Review*, 57(4): 99.

Zhou, B.; Zhao, H.; Puig, X.; Xiao, T.; Fidler, S.; Barriuso, A.; and Torralba, A. 2019. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127: 302–321.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2020. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*.