Collaboration of Experts: Achieving 80% Top-1 Accuracy on ImageNet with 100M FLOPs

Yikang Zhang [1]   Zhuo Chen [1]   Zhao Zhong [1]

Abstract

In this paper, we propose a Collaboration of Experts (CoE) framework to assemble the expertise of multiple networks towards a common goal. Each expert is an individual network with expertise on a unique portion of the dataset, contributing to the collective capacity. Given a sample, delegator selects an expert and simultaneously outputs a rough prediction to trigger potential early termination. For each model in CoE, we propose a novel training algorithm with two major components: weight generation module (WGM) and label generation module (LGM). It fulfills the co-adaptation of experts and delegator. WGM partitions the training data into portions based on delegator via solving a balanced transportation problem, then impels each expert to focus on one portion by reweighting the losses. LGM generates the label to constitute the loss of delegator for expert selection. CoE achieves the state-of-the-art performance on ImageNet, 80.7% top-1 accuracy with 194M FLOPs. Combined with PWLU and CondConv, CoE further boosts the accuracy to 80.0% with only 100M FLOPs for the first time. Furthermore, experiment results on the translation task also demonstrate the strong generalizability of CoE. CoE is hardware-friendly, yielding a 3~6x acceleration compared with existing conditional computation approaches.

1. Introduction

From simple systems to complicated ones, the accomplishment of various tasks relies on the collaboration of multiple individuals. Similarly, a wise combination of models with different properties could yield improved performance on a specific task compared to only deploying one individual

model. There are many approaches for model collaboration, among which ensemble learning (Hansen & Salamon, 1990; Wen et al., 2020; Wenzel et al., 2020) is a popular one. Ensemble learning uses a consensus scheme to decide the collective result by vote. However, it requires multiple forward passes, leading to a significant runtime cost. MIMO (Havasi et al., 2021) draws inspiration from model sparsity (Frankle & Carbin, 2019) and tries to ensemble several subnetworks within one regular network. It only needs one single forward pass of the regular network but is incompatible with compact models. Conditional computation methods (Cheng et al., 2020; Yan et al., 2015) adopt the delegation scheme for model collaboration, conditionally assigning one or several, rather than all models, to make the prediction. Some recently proposed conditional computation methods (Yang et al., 2019; Zhang et al., 2021) have achieved remarkable performance based on dynamic convolution. Nonetheless, they usually have high memory access cost (MAC) and a low degree of parallelism, increasing the real latency (Ma et al., 2018).

Motivated by this, we propose the Collaboration of Experts (CoE) framework to both eliminate the need for multiple forward passes and keep hardware-friendly. CoE consists of one delegator and multiple experts. Firstly, delegator gives a rough prediction and makes the expert selection. If the rough prediction is unreliable, the selected expert will make the refined prediction. Otherwise, the procedure will be early terminated to save FLOPs. Moreover, we only need to load the selected expert into memory, thus keeping the ratio of MAC to FLOPs as a constant. By contrast, dynamic convolution methods (Zhang et al., 2020; 2021) need to load a large number of parameters, namely basis models or experts, to synthesize the input-dependent ones. It enlarges MAC and reduces the degree of parallelism, resulting in a significant deceleration.

To make each model in CoE play its role, we propose a novel training algorithm (as shown in Fig.1) which consists of two major components: weight generation module (WGM) and label generation module (LGM). LGM generates the label (selection label) to constitute the loss of delegator for expert selection. Selection label is a one-hot vector, indicating the suitable expert for each given input. It is obtained by solv-
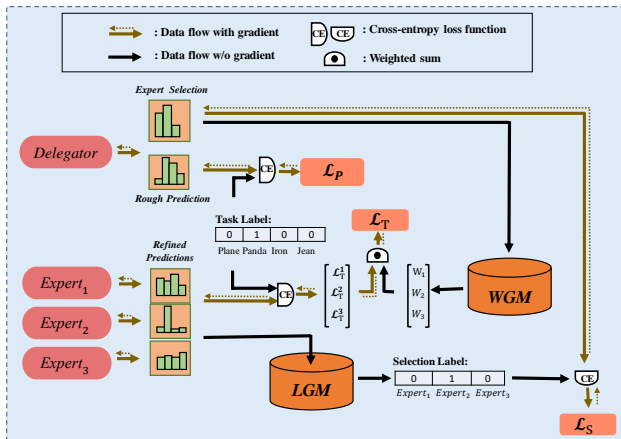
Figure 1. Training framework of CoE with three experts. $\mathcal{L}_T^1$, $\mathcal{L}_T^2$ and $\mathcal{L}_T^3$ are the expert losses, which are reweighted by WGM with weights W_1, W_2 and W_3. LGM generates the label to constitute the loss $\mathcal{L}_S$ for expert selection. $\mathcal{L}_P$ is the cross-entropy loss for the rough prediction of delegator.
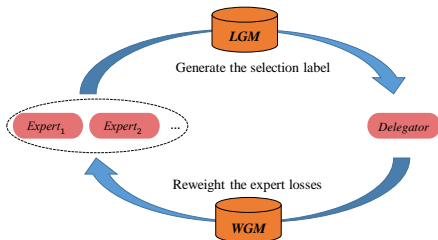

Figure 2. The co-adaptation of experts and delegator. LGM generates the selection label which indicates the suitable expert to supervise delegator for expert selection. WGM partitions the training data into portions based on delegator and impels each expert to focus on one portion by reweighting the losses.

ing a balanced transportation problem (BTP, Shore 1970). Based on delegator, WGM partition the training data into portions by maximizing the summed selection probability via solving BTP as well. Then expert losses are reweighted so that each expert can focus on one portion. As shown in Fig.2, this fulfills the co-adaptation of experts and delegator. The co-adaptation manner makes CoE generalize well to the validation set. Due to the random initialization of experts, selection labels are irregular in the early training stage. Nonetheless, delegator tends to learn generalizable patterns first, since networks learn gradually more complex hypotheses during training (Arpit et al., 2017). Therefore, WGM can partition the training data into portions based on generalizable patterns with delegator as the bridge. It makes selection labels more regular in return, thus delegator avoids overfitting to the irregular labels.

We conduct the main experiments on ImageNet classification task. CoE achieves 78.2/80.7% top-1 accuracy with

only 100/194M FLOPs, while the accuracy for ensembled models (Hansen & Salamon, 1990) is only 79.6% with 920M FLOPs. Compared with the widely-used gate-value-based optimization method (Shazeer et al., 2017; Fedus et al., 2021), our proposed training algorithm improves 1.2% accuracy for CoE, indicating the training effectiveness. Compared with dynamic network approaches, CoE is more hardware-friendly. It not only outperforms the SOTA dynamic method BasisNet which achieves 80.0% accuracy with 198M FLOPs (Zhang et al., 2021), but also accomplishes a 3.1x speedup on hardware. Besides, CoE can be equipped with CondConv and further improve the accuracy to 79.2/81.5% with 102/214M FLOPs. Moreover, we further boost the accuracy to 80.0% with only 100M FLOPs for the first time by using PWLU activation function (Zhou et al., 2021). Experiment results on the translation task also demonstrate the strong generalizability of CoE.

The contributions of this paper can be summarized as follows:

- We propose a collaboration framework named Collaboration of Experts (CoE). The core advantage of it is the inference efficiency. Compared with other conditional computation methods, CoE has low memory access cost and a high degree of parallelism, which are two important factors for real latency.

- We present a novel optimization strategy for CoE that fulfills the co-adaptation of experts and delegator. Experiment results demonstrate its superiority over the widely-used gate-value-based optimization method.

- CoE updates the state-of-the-art on ImageNet for mobile setting, achieving 80.7% top-1 accuracy on ImageNet with less than 200M FLOPs for the first time as far as we know.

- Since the expert selection is done at the model level, CoE can take advantage of existing techniques like conditional convolution and PWLU activation function to push the performance to a new level, namely, 80.0% accuracy on ImageNet with only 100M FLOPs.

2. Related Work

2.1. Ensemble Learning and Model Selection

Ensemble learning (Hansen & Salamon, 1990) aims at combining the predictions from several models to get a more robust one. Some recently proposed literatures (Wen et al., 2020; Wenzel et al., 2020) demonstrate that significant gains can be achieved with negligible additional parameters compared to the original model. However, these methods still require multiple (typically, 4-10) forward passes for

prediction, leading to a significant runtime cost. Differently, CoE utilizes a delegator to select only one expert for the refined prediction, thus at most two forward passes are needed. MIMO (Havasi et al., 2021) draws inspiration from model sparsity (Frankle & Carbin, 2019) and holds the view that multiple independent subnetworks can be concurrently trained within one regular network because of the heavy parameter redundancy. Therefore, those subnetworks can be ensembled with a single forward pass of the regular model. However, MIMO cannot be applied to compact models which have already been pruned or the ones constructed by AutoML methods (Cai et al., 2020; Zhong et al., 2018). It is because these models have few redundant parameters. By contrast, CoE is free from the compactness of experts since expert selection is done at the model level. Recently, some works about model selection are proposed (Li et al., 2021b; You et al., 2021). These methods are concerned with ranking a number of pre-trained models and finding the one transfers best to a downstream task of interest. Therefore, they select models task-wisely. By contrast, CoE aims at improving the task performance via selecting the most suitable expert for each sample instance-wisely.

2.2. Dynamic Networks

Dynamic networks achieve high performance with low computation cost by conditionally varying the network parameters (Zhang et al., 2020; Yang et al., 2019) or network architectures (Yuan et al., 2020). HD-CNN (Yan et al., 2015) and HydraNet (Mullapudi et al., 2018) select branches based on the category, they cluster all categories into n groups, where n is the number of branches. While CoE learns the model selection pattern automatically, it can be based on any property, rather than limited to the category. MoE (Shazeer et al., 2017) and Switch Transformer (Fedus et al., 2021) select experts at the layer level with a router. The output feature of each expert will be scaled with the gate-value predicted by router, thus router becomes trainable. This gate-value-based optimization manner is heuristic while CoE trains the delegator more reasonably. Since expert selection for CoE is done across models, we can use protocols like True Class Probability (TCP, Corbière et al. 2019) to measure the suitability of each expert without bias. Based on expert suitabilities, the labels to supervise delegator for expert selection can be obtained. Additionally, CoE takes more advantage of conditional computation as the whole network is selected, rather than only some particular layers. The recently proposed Dynamic Convolution methods (Zhang et al., 2020; Yang et al., 2019; Chen et al., 2020) share the similar idea and achieve remarkable performance with low FLOPs but high latency. It is because these methods need to load many basis models or experts to synthesize the dynamic parameters, causing high MAC and low degree of parallelism (Ma et al., 2018). By contrast, CoE only needs to load the selected

expert into memory, avoiding these problems. Finally yet importantly, batch processing is an important method to enhance the degree of parallelism. Because of the input-dependent parameters (Shazeer et al., 2017; Zhang et al., 2021) or architectures (Yuan et al., 2020), these methods cannot process samples in batch. Differently, CoE supports batch processing because the number of experts is limited and each one of them corresponds to many test samples.

3. Method

CoE consists of a delegator and n experts, a total of $n + 1$ individual neural networks. Given a sample, delegator will select an expert and simultaneously output a rough prediction to trigger potential early termination. Since the inference of delegator is conducted all the time, we prefer to make delegator more lightweight than expert. Delegator consists of three modules: feature extractor, task predictor and expert selector as shown in Fig.3. Based on the feature derived from feature extractor, task predictor and expert selector output the probabilities for classification and expert selection respectively. In the following subsections, we will describe the inference procedure and training strategy of CoE comprehensively. The number of samples and experts are denoted as m and n.
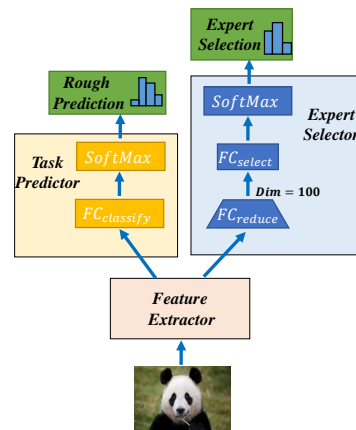

Figure 3. The architecture of delegator. Task predictor consists of one fully connected layer and a SoftMax layer. Expert selector consists of two fully connected layers with the hidden dim as 100, followed by a SoftMax layer.

3.1. Inference Procedure

CoE firstly uses delegator to obtain the rough prediction and determine the selected expert for each sample. Afterward, *Maximum Class Probability* (MCP, Corbière et al. 2019) of rough prediction is calculated. It is the probability of predicted class. Then, the final recognition results for samples with MCP larger than a given threshold τ are derived from the rough predictions. Other samples are partitioned into

n groups based on which expert is selected. Subsequently, batch processing can be conducted within each group to obtain refined predictions. This procedure is shown in Fig.4. The averaged FLOPs/Instance of CoE ranges from F_D to $F_D + F_E$ by varying τ from 0 to 1, F_D and F_E are FLOPs of delegator and experts. Therefore, the value of τ is directly determined by target FLOPs.
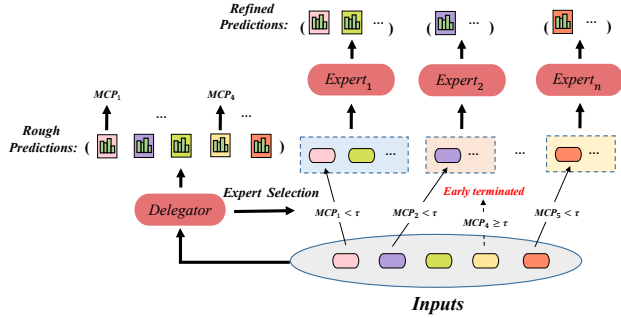

Figure 4. The inference procedure of CoE. MCP is the probability of predicted class. τ is the threshold for early termination.

3.2. Label Generation Module (LGM)

Since expert selection for CoE is done across models, we can measure the suitability of each expert without bias. Based on this, labels to supervise delegator for expert selection can be obtained. This label-based training method for delegator is more reasonable than the widely-used gate-value-based method (Shazeer et al., 2017; Fedus et al., 2021) which enables the training of routing function by using its predicted gate-value to scale the output of each expert. Next, we firstly introduce how to measure the suitability of each expert, then illustrate how to obtain the selection label.

Model accuracy can be measured by *True Class Probability* (TCP, Corbière et al. 2019):

$$\text{TCP}_{j,k} = P(Y = y_j | x_j, \text{Expert}_k), \quad (1)$$

where, x_j is the j-th sample, Y and y_j are the predicted and true class. But accuracy is not the only factor for suitability. For example, when models are of different sizes, the larger one usually has a higher TCP. But it may not be more suitable, due to the large inference cost. Given that our concern is not the optimization of network architecture, we can suppose no expert is superior to others (*No Superiority Assumption*, NSA). Motivated by NSA, we leverage the standardized TCP as the metric for sutability:

$$S_{j,k} = \frac{\text{TCP}_{j,k} - Mean(\text{TCP}_{:,k})}{Std(\text{TCP}_{:,k})}, \quad (2)$$

where, $Mean(\text{TCP}_{:,k})$ and $Std(\text{TCP}_{:,k})$ are mean value and standard deviation for TCPs of Expert_k on m samples.

Selection labels can be denoted by a binary matrix $L_{m \times n}$, where each row is a selection label. According to NSA, each expert should be assigned the same number of samples, thus the sum of each colum vector of $L_{m \times n}$ should be same, i.e. $\sum_j L_{j,k} = \frac{m}{n}$ for $k = 1, ..., n$. Therefore, $L_{m \times n}$ can be obtained by maximizing $\sum_{j,k} S_{j,k} * L_{j,k}$:

$$\begin{aligned} \min \quad & \sum_{j,k} -S_{j,k} * L_{j,k} \\ s.t. \quad & L_{j,k} \in \{0,1\}, \quad \sum_k L_{j,k} = 1, \quad \sum_j L_{j,k} = \frac{m}{n} \end{aligned} \quad (3)$$

This problem can be modeled as the balanced transportation problem (BTP, Shore 1970), where each sample is a supply source with a supply of one, each expert is a demand source with a demand of m/n. $-S_{j,k}$ is the per-unit transportation cost from the j-th supply source to the k-th demand source. We solve this problem via Vogel approximation method (VAM, Shore 1970) as introduced in Appendix A.1, which is a short-cut approach to invariably obtain a good solution.

3.3. Weight Generation Module (WGM)

To maximize the collective capacity of CoE, the dataset needs to be partitioned into portions then each expert only focuses on one portion. This is achieved by WGM which reweights the losses of experts. The partition can be indicated by an assignment matrix $A_{m \times n}$, with one-hot row vectors. $A_{j,k} = 1$ means the j-th sample x_j is assigned to the k-th expert, thus the loss weight for Expert_k gets larger than other experts on x_j.

A naive partition can be based on expert suitability, namely, partitioning the training data with selection labels $L_{m \times n}$. However, it results in a poor generalization to delegator as shown in Fig.5a. Assuming Expert_k is suitable on a sample x_j, thus $A_{j,k} = L_{j,k} = 1$. Due to $A_{j,k} = 1$, the loss weight for Expert_k gets larger than other experts on x_j, making Expert_k more suitable in return. Therefore selection labels cannot be updated and the irregularity for them caused by random initialization will be preserved. Consequently, delegator gradually overfits to irregular labels, yielding a poor generalization. This is also verified in Section 4.4, where the expert-suitability-based partition results in a terrible performance for $\text{CoE}^{WGM^\star}$.

Since networks learn gradually more complex hypotheses during training (Arpit et al., 2017), delegator tends to learn generalizable patterns first. Therefore, the partition can be based on generalizable patterns with delegator as the bridge, namely, partition based on the output of delegator. In this way, selection labels get more regular in return due to the reweighting of expert losses. As shown in Fig.5b, delegator avoids overfitting to the irregular labels.

Delegator outputs a probability matrix $P_{m \times n} \in R^{m \times n}$,
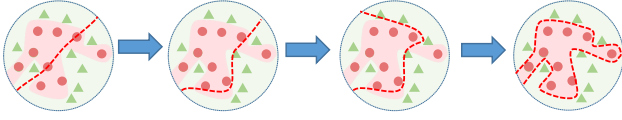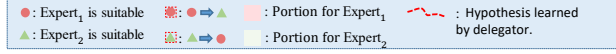
(a) Partition the training data based on expert suitability, which makes delegator overfit to irregular selection labels.
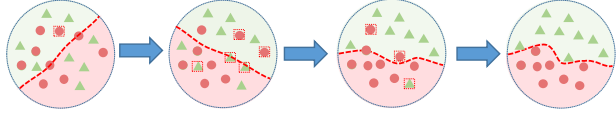
(b) Partition the training data based on delegator, which fulfills the co-adaptation of experts and delegator, avoiding the overfitting problem of delegator.

Figure 5. Demo with two experts of the training process. The training samples are denoted as green triangles or pink circles, where pink circle indicates $Expert_1$ is more suitable for it than $Expert_2$. During training, which expert is more suitable may change, we mark it with a dashed bounding square. As mentioned, the training data need to be partitioned into two portions then each expert only focuses on one portion. We use pink background to represent the portion for $Expert_1$ and the green one for $Expert_2$. Delegator learns hypothesis to select expert, it is denoted as red dashed curve.

whose element $P_{j,k} \in [0,1]$ represents the probability of selecting the k-th expert on the j-th sample. As analyzed above, it is better to partition the training data based on $P_{m \times n}$, thus $A_{m \times n}$ is obtained by maximizing $\sum_{j,k} P_{j,k} * A_{j,k}$. Moreover, according to NSA, the number of samples assigned to each expert should be same, i.e. $\sum_j A_{j,k} = m/n$. Thus, $A_{m \times n}$ is optimized by:

$$
\begin{aligned}
\min \quad & \sum_{j,k} -P_{j,k} * A_{j,k} \\
s.t. \quad & A_{j,k} \in \{0,1\}, \quad \sum_k A_{j,k} = 1, \quad \sum_j A_{j,k} = \frac{m}{n}
\end{aligned}
\tag{4}
$$

This problem can also be modeled as BTP, and solved via VAM as described in section 3.2. In the early training stage, the models in CoE are underfitted. Thus we cannot trust $A_{m \times n}$ and need to make the gap between loss weights for different experts smaller to warm up. We achieve this by smoothing $A_{m \times n}$ to $\overline{A}_{m \times n}$ with Eq.5, where α grows linearly from 0.2 to 0.8 with the training going on,

$$
\overline{A}_{j,k} = \begin{cases} \alpha + \frac{1-\alpha}{n}, & if \ A_{j,k} = 1 \\ \frac{1-\alpha}{n}, & if \ A_{j,k} = 0 \end{cases}.
\tag{5}
$$

Finally, the output of WGM (i.e. $W_{m \times n}$) is obtained by

normalizing $\overline{A}_{m \times n}$ with the coefficient $\mathcal{Z} = \sum_j \overline{A}_{j,k} = \frac{m}{n}$:

$$
W_{j,k} = \frac{\overline{A}_{j,k}}{\mathcal{Z}}.
\tag{6}
$$

3.4. Training Details

The training framework of CoE is shown in Fig.1, which consists of three major losses: $\mathcal{L}_P$, $\mathcal{L}_T$ and $\mathcal{L}_S$.

$\mathcal{L}_P$ is the cross-entropy loss for the rough prediction of delegator. To avoid the repeated training of delegator, we use $\mathcal{L}_P$ to train the feature extractor and task predictor (Fig.3) first of all. Then these two modules are fixed, only expert selector and experts are jointly optimized with $\mathcal{L}_{Total} = \eta * \mathcal{L}_S + \mathcal{L}_T$, η is set as 0.8 in this paper.

$\mathcal{L}_S$ is used to optimize the expert selector. Based on the selection label $L_{j,:}$, we can get the cross-entropy loss $\mathcal{L}_S^j$ for the j-th sample. Because the final recognition result of CoE is not always sensitive to expert selection, $\{\mathcal{L}_S^j | j = 1, \ldots, m\}$ should be attached various importance. For example, when experts have similar suitabilities (Eq.2) on the j-th sample, expert selection will have little influence to final performance of CoE, therefore the weight for $\mathcal{L}_S^j$ gets smaller. Considering the similarity of suitabilities can be measured by the standard deviation $Std(S_{j,:})$, we set the loss weight for $\mathcal{L}_S^j$ as $v_j = \frac{Std(S_{j,:})}{\sum_i Std(S_{i,:})}$. Finally,

$$
\mathcal{L}_S = \sum_j v_j * \mathcal{L}_S^j.
\tag{7}
$$

$\mathcal{L}_T$ is used to optimize the experts. Based on the class labels of m samples, we can get $m \times n$ cross-entropy losses $\{\mathcal{L}_T^{j,k} | j = 1, \ldots, m; k = 1, \ldots, n\}$, where $\mathcal{L}_T^{j,k}$ is the cross-entropy loss for the k-th expert on the j-th sample. Then $\mathcal{L}_T$ is obtained by the weighted sum of $\mathcal{L}_T^{j,k}$ with weights $W_{j,k}$ output by WGM:

$$
\mathcal{L}_T = \sum_{j,k} W_{j,k} * \mathcal{L}_T^{j,k}.
\tag{8}
$$

We use either four or sixteen experts in this paper. When using sixteen experts, we decompose the task into four sub-tasks, each of which involves four experts as described in Appendix A.2. This reduces the memory cost for training.

4. Experiments

We conduct the main experiments on ImageNet classification task. After comparing with some popular efficient models, we verify the superiority of CoE over the existing model collaboration methods: model ensemble and category-based model selection. Afterward, the effectiveness of training
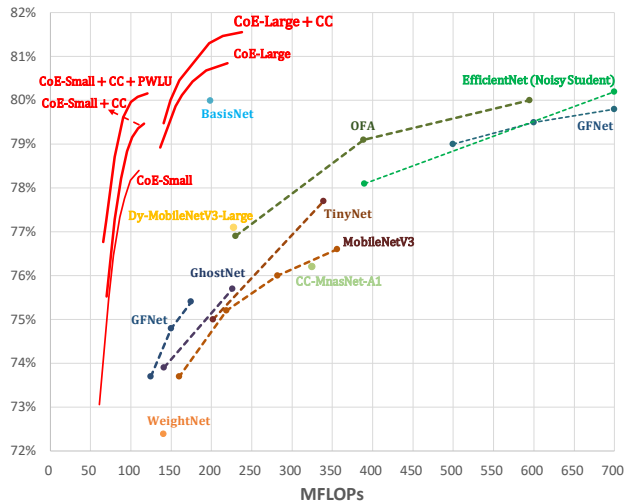
Figure 6. Top-1 Accuracy v.s. FLOPs for CoE on ImageNet. "CC" means CondConv and 'PWLU' is an activation function.

Table 1. Compare CoE with some efficient models on ImageNet.

Method	FLOPs	TOP-1 Acc
WeightNet (2020)	141M	72.4%
DS-MBNet-M[†‡](2021a)	319M	72.8%
GhostNet 1.0x (2020a)	141M	73.9%
MobileNetV3-Large (2019)	219M	75.2%
OFA-230 (2019)	230M	76.9%
TinyNet-A (2020b)	339M	77.7%
CondConv-EfficientNet-B0 (2019)	413M	78.3%
GFNet (2020)	400M	78.5%
CoE-Small	100M	78.2%
CoE-Small + CondConv	102M	79.2%
CoE-Small + CondConv + PWLU	100M	80.0%
BasisNet (2021)	198M	80.0%
OFA-595 (2019)	595M	80.0%
EfficientNet-B2 (2019)	1.0B	80.1%
EfficientNet-B1(Noisy Student) (2020)	700M	80.2%
BasisNet (2021)	290M	80.3%
FBNetV3-C (2020)	557M	80.5%
BasisNet + CondConv (2019)	308M	80.5%
CoE-Large	194M	80.7%
CoE-Large + CondConv	214M	81.5%

strategy is analyzed by the comparison with the widely-used gate-value-based training method and the elaborated ablations. Moreover, we try to generalize CoE to the translation task and re-evaluate CoE using Reassessed Labels (ReaL) (Beyer et al., 2020). Finally, we try to analyze the reasonability of learned expert selection patterns. Statistics on referenced baselines in section 4.2.1&4.2.2 are directly cited from original papers, others are implemented with the following setting unless otherwise stated.

4.1. Implementation Details

We conduct experiments with two settings: **CoE-Small** and **CoE-Large**. For CoE-Small, we take TinyNet-E (Han et al., 2020b) with 24M FLOPs as the feature extractor of delegator by removing the last fully connected layer. We use OFA-110 (Cai et al., 2020) with 110M FLOPs as the expert. For CoE-Large, MobileNetV3-Small (Howard et al., 2019) with 56M FLOPs is adopted to construct the delegator by analogy. We use OFA-230 with 230M FLOPs as the expert. We have also tried to introduce CondConv (Zhang et al., 2020) and PWLU activation fuction (Zhou et al., 2021) to achieve the extreme performance. More details are illustrated in Appendix B.1.

4.2. Results and Analysis

4.2.1. ACCURACY AND COMPUTATION COST

By varying the threshold τ of early termination, the accuracy curves for CoE are obtained. We show them in Fig. 6, then pick out a point from each curve to compare with some efficient models in Table 1. CoE achieves 78.2% and 80.7% accuracy with 16 experts and the averaged FLOPs/Instance

are 100M and 194M respectively. Compared with OFA, CoE reduces the FLOPs from 230M to 100M and from 595M to 194M, with better top-1 accuracy. Though dynamic networks like GFNet, CondConv and BasisNet are more efficient than traditional networks, CoE still has significantly higher accuracy with smaller FLOPs. Compared with these approaches, CoE improves the accuracy by 2.2/2.4/0.7% respectively. When combined with CondConv, CoE achieves 79.2% and 81.5% accuracy with only 102M and 214M FLOPs, indicating that CoE is complementary to dynamic networks like CondConv. On the contrary, as CondConv and BasisNet share similar essence, namely using a group of basis to dynamically synthesize the input-dependent convolution kernel, the combination of them only arouses little collaborative benefit with the accuracy as 80.5%. Moreover, CoE achieves 80.0% accuracy with 100M FLOPs for the first time by further using PWLU.

4.2.2. INFERENCE SPEED AND MEMORY COST

The core advantage of CoE is the inference efficiency. To verify this advantage, we analyze the inference latency on hardware. The experiments are conducted on CPU platform (Intel(R) Xeon(R) CPU E5-2699 v4 @ 2.20GHz) with PyTorch version as 1.8.0. We report the averaged latency on the ImageNet validation set in Table 2. We notice the discrepancy between FLOPs and real speed. For example, OFA-230 has 1.6x FLOPs compared with GhostNet 1.0x, but the speed is 1.2x faster. Moreover, this discrepancy can be enlarged by CondConv. CondConv-EfficientNet-B0 has similar FLOPs with the original EfficientNet-B0, but the

speed is 1.7x slower. BasisNet synthesizes the dynamic parameters all at once, rather than the "layer by layer" fashion like CondConv, thus is more efficient. However, it still needs to load a large number of parameters for this synthesis, which brings a large MAC. This is why CoE (16 experts) can reduce 14.09% latency than BasisNet even when the mini-batch size is one. Finally yet importantly, BasisNet and CondConv do not support batch processing, while CoE (16 experts) can take advantage of it to further achieve a 3.1/6.1x speedup compared with them. We analyze the memory cost from two perspectives: the number of parameters and MAC. As can be seen from Table 2, the accuracy of CoE-Large (4 experts) is no worse than BasisNet and CondConv-EfficientNet-B0 when using similar parameters. Besides, the averaged MAC/Instance of CoE is much smaller than theirs.

4.2.3. ANALYSIS OF THE TRAINING COST

To achieve superior performance with fewer inference FLOPs and latency, CoE may consume more training time. For example, the applying of CoE (4 experts) on OFA-230 improves the accuracy from 78.0% to 79.9%, but at the expense of a 2.2x training cost. To verify whether the improvement still exists with similar training cost, we get a series of accuracies by varying the number of training epochs as shown in Fig.7, where 32 GPUs (Tesla-V100-PCIe-16GB) are used. It is seen that CoE boosts the performance from 78.3% to 79.9% even when the training cost is similar.
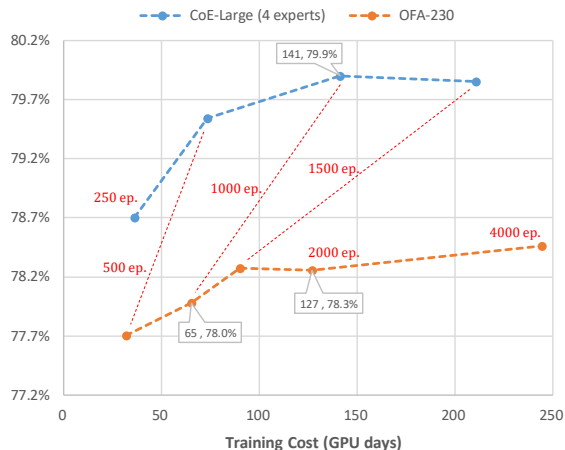

Figure 7. Accuracy v.s. training cost for OFA-230 and CoE-Large (4 experts). "xx ep." means the number of training epochs is "xx".

4.3. Comparison with Existing Approaches

4.3.1. COMPARISON WITH MODEL ENSEMBLE

We train four OFA-230 models with different initialization seeds as shown in Table 3. The random initialization usually causes minor variety in accuracy for ImageNet classifica-

tion but leads models to fall into different local minima, yielding the diversity of output. This diversity enables the ensembled models to achieve an improvement of 1.6% in accuracy. We adopt the naive ensemble here, i.e. averaging the output of each model. As shown in Wen et al. (2020); Chen et al. (2019), though simple, naive ensemble has a competitive performance in terms of accuracy. However, CoE still achieves 1.1% higher accuracy than it. Recently proposed ensemble methods (Havasi et al., 2021) mainly focus on reducing the computation cost with a little drop of accuracy than naive ensemble, but the computation cost is always larger than the one of a base model. By contrast, CoE reduces the FLOPs to $0.84\times$ of the base model, indicating the superiority of CoE in terms of FLOPs as well.

4.3.2. COMPARISON WITH CATEGORY-BASED METHOD

HD-CNN and HydraNets select branches based on the category. Despite their methods are originally designed to select a specific block, we apply them to the model level. To select expert based on category, the categories should be partitioned into n groups, where n is the number of experts. We try two schemes: random partition and clustering-based partition. Then, expert can be selected according to the rough prediction of delegator. During the training procedure, we also reweight losses of each expert based on the assignment matrix $A_{m\times n}$ with Eq.5&6. Here, $A_{m\times n}$ is obtained directly based on the rough prediction. The results with **4 experts** are shown in Table 4, demonstrating a better collaboration pattern is learned by CoE.

4.3.3. COMPARISON WITH GATE-VALUE-BASED TRAINING METHOD

MoE and Switch Transformer adopt the gate-value-based training method for routing function. They enable the training of router by using its predicted gate-value to scale the output of each expert. This optimization manner is heuristic while CoE trains the delegator more reasonably. Since expert selection for CoE is done across models, we can measure the suitability of each expert without bias. Thereafter, selection labels can be obtained to supervise delegator. Despite the gate-value-based method is originally designed to select a specific layer, we apply it to the model level. We compare with both the soft gate-value and the hard gate-value. For hard gate-value, it is a one-hot vector generated by replacing the softmax in expert selector (Fig.3) with gumbel softmax (Jang et al., 2017). The results with **4 experts** are shown in Table 5, where CoE achieves better performance, indicating the effectiveness of our training method.

4.4. Ablation Studies for CoE

We have conducted elaborated ablations, including ablations for each element of our proposed training method,

Table 2. CPU latency and memory cost for different methods.

Models	CPU Latency/Instance (ms)		FLOPs	MAC	Params	Accuracy
	Batchsize=1	Batchsize=64				
MobileNetV3-Small	14.77	4.18	56M	2.5M	2.5M	67.4%
GhostNet 1.0x	39.91	16.50	141M	5.2M	5.2M	73.9%
TinyNet-B	34.58	19.44	202M	3.7M	3.7M	75.0%
MobileNetV3-Large	31.55	18.43	219M	5.4M	5.4M	75.2%
GhostNet 1.3x	43.94	29.70	226M	7.3M	7.3M	75.7%
OFA-230	33.52	15.21	230M	5.8M	5.8M	76.9%
EfficientNet-B0	49.12	35.21	391M	5.3M	5.3M	77.2%
TinyNet-A	45.76	23.71	339M	5.1M	5.1M	77.7%
CondConv-EfficientNet-B0	81.81	-	413M	24.0M	24.0M	78.3%
BasisNet	40.61	-	198M	24.9M	24.9M	80.0%
CoE-Large (4 experts)	38.67	15.02	220M	6.6M	25.7M	79.9%
CoE-Large (16 experts)	34.89	13.30	194M	6.0M	95.3M	80.7%

Table 3. Comparison with model ensembe.

Method		FLOPs	Acc.
OFA-230	Seed1	230M	78.1%
	Seed2	230M	78.0%
	Seed3	230M	78.1%
	Seed4	230M	78.0%
	Ensemble	920M	79.6%
CoE-Large	4 Experts	194M	79.8%
	16 Experts	194M	80.7%

Table 4. Comparison with category-based selection. "RP" and "CBP" means Random Partition and Clustering-Based Partition.

Method		FLOPs	Top-1 Acc.
Category-Based	RP	220M	78.3%
	CBP	220M	77.5%
CoE-Large	-	220M	79.9%

ablations for the training tricks, ablations for expert number and early termination. We mainly introduce the ablations for our proposed training method here, others are discussed in Appendix B.2.

CoE consists of 2 major components: LGM and WGM. Apart from directly removing one component, we also try to alter some elements inside them. We propose several modified versions of CoE for ablation as below:

- CoE^{LGM}: Remove LGM from CoE. Thus, CoE collapses to a single expert with delegator to trigger the

Table 5. Comparison with gate-value-based traning method.

Method		FLOPs	Top-1 Acc.
Gate-Value-Based	Soft Gate-Value	220M	78.7%
	Hard Gate-Value	220M	78.9%
CoE-Large	-	220M	79.9%

early termination.

- CoE$^{LGM^\star}$: Abandon the refining of suitability criterion (Eq.2) and remove the "$\sum_j L_{j,k} = m/n$"constraint in Eq.3. So that $L_{m\times n}$ neglects the *No Superiority Assumption* (NSA).

- CoE^{WGM}: Remove WGM from CoE. Thus, losses of experts have identical weights for each sample.

- CoE$^{WGM^\star}$: WGM partitions the training data based on expert suitability, thus the assignment matrix $A_{m\times n}$ in WGM equals to the output matrix $L_{m\times n}$ of LGM.

- CoE$^{WGM^\circ}$: Remove the "$\sum_j A_{j,k} = m/n$" constraint in Eq.4 and abandon the smoothing of $A_{m\times n}$ (Eq.5). Thus WGM neglects the NSA.

- CoE$^{WGM^\bullet}$: Abandon the progressive sharpening for $A_{m\times n}$ in WGM. Specifically, set α in Eq.5 as a constant 0.8, instead of linearly increasing it.

- CoE^{SR}: Abandon the reweighting for losses of expert selection ($\mathcal{L}_S^j$), namely set v_j in Eq.7 as a constant $\frac{1}{m}$.

The results for those CoE versions with the CoE-Large setting and 4 experts are shown in Table 6, which demonstrates the significance of each element of the training method.

Table 6. Ablations for each component of the training method.

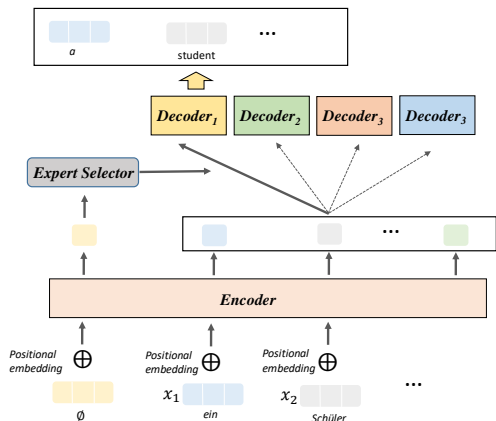| Method	Experts	FLOPs	Acc.
CoE-Large	4	220M	79.9%
CoE-Large^{LGM}	4	220M	78.0%
CoE-Large$^{LGM^\star}$	4	220M	79.4%
CoE-Large^{WGM}	4	220M	78.1%
CoE-Large$^{WGM^\star}$	4	220M	77.0%
CoE-Large$^{WGM^\circ}$	4	220M	79.2%
CoE-Large$^{WGM^\bullet}$	4	220M	79.4%
CoE-Large^{SR}	4	220M	79.5%

Figure 8. The architecture of CoE-Transformer. ϕ is a constant token, whose feature is used for expert selection.

4.5. Analysis of the Generalization

To verify the generalizability, we conduct two extra experiments: **_generalizing CoE to translation task_** and using Re-assessed Labels (ReaL) (Beyer et al., 2020) to **_re-evaluate CoE_**. We mainly introduce the first one here, another one are discussed in Appendix B.3.

To generalize CoE to translation task, we build a CoE-Transformer model based on Transformer (base model) (Vaswani et al., 2017). CoE-Transformer has four decoders, given a sentence, one decoder will be selected to decode the features extracted by encoder. To select the decoder, an extra constant token ϕ is added at the beginning of each sentence, whose feature extracted by encoder is input to the expert selector (Fig.3) for expert selection. During training, the TCP of a sentence is obtained by averaging the TCPs of each token. The architecture of CoE-Transformer are shown in Fig. 8.

Following (Vaswani et al., 2017; Ott et al., 2019), CoE-Transformer is trained on the standard WMT 2014 English-German dataset. As mentioned, an extra token will be added to this vocabulary. We adopt the same training and evaluating setting as (Ott et al., 2019), more details are shown in Appendix B.4. From Table 7 we can see, CoE-Transformer outperforms Transformer (base model) by a large margin and achieves similar performance as Transformer (big) with much less MAC and parameters.

Table 7. The BLEU scores on newstest2014 (English-to-German).

Model	MAC	Parameters	BLEU
Transformer (base)	62.4M	62.4M	28.1 (2019)
Transformer (big)	213.0M	213.0M	29.3 (2019)
CoE-Transformer	62.5M	138.2M	29.4

4.6. Analysis of the Learned Expert Selection Patterns

We also conduct experiments to analyze the expert selection patterns of CoE and find them quite reasonable. When experts have different architectures, the delegator tends to assign easy samples to smaller experts and complex samples to heavier experts. When experts share the same architecture, delegator learns the expert selection patterns automatically, it can be based on any property (e.g. whether humans are contained), rather than limited to the category. We introduce the experiment when experts have various architectures here, more details are illustrated in Appendix B.5.

Considering TCP (Corbière et al., 2019) measures the complexity of a given sample if the inference model is fixed, namely, the more complex is the sample, the smaller TCP will be. We can analyze the relationship between sample complexity and expert selection. We take four architectures searched via OFA (Cai et al., 2020) as the experts, i.e. OFA-110, OFA-163, OFA-230 and OFA-595. OFA-xx indicates the FLOPs is xx. The delegator is also MobileNetV3-small as described in section 4.1. We obtain the TCP value for each sample based on the delegator. We count the selection probability for each expert at different TCP values on the validation set. As shown in Fig.9, the selection probability for heavier model increases with the input sample getting more complex (with the decrease of TCP). It demonstrates that CoE can learn reasonable expert selection patterns automatically.
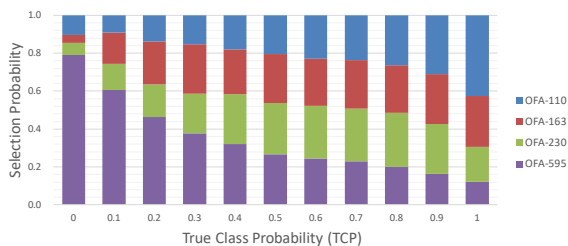

Figure 9. Selection probabilities for each expert at different TCP values. The selection probability for heavier model increases with the input sample getting more complex (with the decrease of TCP)

5. Conclusion

We propose a CoE framework to pool together the expertise of multiple networks towards a common aim. Experiments in this paper demonstrate the superiority of CoE on both accuracy and real speed. We also analyze the collaboration patterns and find them have interpretability. In the future, CoE will be extended to the trillion parameters level. Meanwhile, we will try to implement CoE to more tasks and verify its compatibility with quantification and other technologies. Besides, CoE has the potential to solve problems of lifelong learning by updating experts.

References

Arpit, D., Jastrzębski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., and Lacoste-Julien, S. A closer look at memorization in deep networks. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 233–242. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/arpit17a.html.

Beyer, L., Hénaff, O. J., Kolesnikov, A., Zhai, X., and van den Oord, A. Are we done with imagenet? *CoRR*, abs/2006.07159, 2020.

Cai, H., Gan, C., Wang, T., Zhang, Z., and Han, S. Once-for-all: Train one network and specialize it for efficient deployment. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HylxE1HKwS.

Chen, Q., Zhang, W., Yu, J., and Fan, J. Embedding complementary deep networks for image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Chen, Y., Dai, X., Liu, M., Chen, D., Yuan, L., and Liu, Z. Dynamic convolution: Attention over convolution kernels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11030–11039, 2020.

Cheng, A.-C., Lin, C. H., Juan, D.-C., Wei, W., and Sun, M. Instanas: Instance-aware neural architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3577–3584, 2020.

Corbière, C., THOME, N., Bar-Hen, A., Cord, M., and Pérez, P. Addressing failure prediction by learning model confidence. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/757f843a169cc678064d9530d12a1881-Paper.pdf.

Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., and Le, Q. V. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 113–123, 2019.

Dai, X., Wan, A., Zhang, P., Wu, B., He, Z., Wei, Z., Chen, K., Tian, Y., Yu, M., Vajda, P., et al. Fbnetv3: Joint architecture-recipe search using neural acquisition function. *arXiv preprint arXiv:2006.02049*, 2020.

Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJl-b3RcF7.

Gontijo-Lopes, R., Smullin, S., Cubuk, E. D., and Dyer, E. Tradeoffs in data augmentation: An empirical study. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=ZcKPWuhG6wy.

Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., and Xu, C. Ghostnet: More features from cheap operations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1577–1586, 2020a. doi: 10.1109/CVPR42600.2020.00165.

Han, K., Wang, Y., Zhang, Q., Zhang, W., Xu, C., and Zhang, T. Model rubik's cube: Twisting resolution, depth and width for tinynets. In *NeurIPS*, 2020b.

Hansen, L. K. and Salamon, P. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12(10):993–1001, 1990.

Havasi, M., Jenatton, R., Fort, S., Liu, J. Z., Snoek, J., Lakshminarayanan, B., Dai, A. M., and Tran, D. Training independent subnetworks for robust prediction. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=OGg9XnKxFAH.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1314–1324, 2019.

Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. Deep networks with stochastic depth. In *European conference on computer vision*, pp. 646–661. Springer, 2016.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017.

Li, C., Wang, G., Wang, B., Liang, X., Li, Z., and Chang, X. Dynamic slimmable network. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8607–8617, June 2021a.

Li, Y., Jia, X., Sang, R., Zhu, Y., Green, B., Wang, L., and Gong, B. Ranking neural checkpoints. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2663–2673, June 2021b.

Ma, N., Zhang, X., Zheng, H.-T., and Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 116–131, 2018.

Ma, N., Zhang, X., Huang, J., and Sun, J. Weightnet: Revisiting the design space of weight networks. *CoRR*, abs/2007.11823, 2020.

Mullapudi, R. T., Mark, W. R., Shazeer, N., and Fatahalian, K. Hydranets: Specialized dynamic architectures for efficient inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8080–8089, 2018.

Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., and Auli, M. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Shore, H. H. The transportation problem and the vogel approximation method. *Decision Sciences*, 1(3-4):441–457, 1970.

Tan, M. and Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pp. 6105–6114. PMLR, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *NIPS*, pp. 6000–6010, 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.

Wang, Y., Lv, K., Huang, R., Song, S., Yang, L., and Huang, G. Glance and focus: a dynamic approach to reducing spatial redundancy in image classification. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 2432–2444. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1963bd5135521d623f6c29e6b1174975-Paper.pdf.

Wen, Y., Tran, D., and Ba, J. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=Sklf1yrYDr.

Wenzel, F., Snoek, J., Tran, D., and Jenatton, R. Hyperparameter ensembles for robustness and uncertainty quantification. *arXiv preprint arXiv:2006.13570*, 2020.

Xie, Q., Luong, M.-T., Hovy, E., and Le, Q. V. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10687–10698, 2020.

Xie, S., Girshick, R., Dollar, P., Tu, Z., and He, K. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Yan, Z., Zhang, H., Piramuthu, R., Jagadeesh, V., DeCoste, D., Di, W., and Yu, Y. Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pp. 2740–2748, 2015.

Yang, B., Bender, G., Le, Q. V., and Ngiam, J. Condconv: Conditionally parameterized convolutions for efficient inference. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

You, K., Liu, Y., Wang, J., and Long, M. Logme: Practical assessment of pre-trained models for transfer learning. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12133–12143. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/you21b.html.

Yuan, K., Li, Q., Chen, D., Zhou, A., and Yan, J. Dynamic graph: Learning instance-aware connectivity for neural networks. *arXiv preprint arXiv:2010.01097*, 2020.

Zhai, X., Oliver, A., Kolesnikov, A., and Beyer, L. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Zhang, M., Chu, C.-T., Zhmoginov, A., Howard, A., Jou, B., Zhu, Y., Zhang, L., Hwa, R., and Kovashka, A. Basisnet: Two-stage model synthesis for efficient inference. *arXiv preprint arXiv:2105.03014*, 2021.

Zhang, Y., Zhang, J., Wang, Q., and Zhong, Z. Dynet: Dynamic convolution for accelerating convolutional neural networks. *arXiv preprint arXiv:2004.10694*, 2020.

Zhong, Z., Yan, J., Wu, W., Shao, J., and Liu, C.-L. Practical block-wise neural network architecture generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Zhou, Y., Zhu, Z., and Zhong, Z. Learning specialized activation functions with the piecewise linear unit. *CoRR*, abs/2104.03693, 2021.

A. Extra Details for Method

A.1. Introduction of Vogel Approximation Method (VAM)

In weight generation module (WGM) and label generation module (LGM), we need to solve the balanced transportation problem (BTP, (Shore, 1970)) via Vogel approximation method (VAM, (Shore, 1970)). We will introduce it in this section with the number of samples and experts as m and n respectively.

The BTP involved in WGM and LGM has m supply sources, each of which is denoted as $Silo_j$ with a supply of one, as well as n demand sources, each of which is denoted as $Mill_k$ with a demand of $\frac{m}{n}$. $C_{j,k}$ is the per-unit transportation cost from $Silo_j$ to $Mill_k$. Specifically, $C_{j,k} = -P_{j,k}$ in WGM and $C_{j,k} = -S_{j,k}$ in LGM. To make it clear, we illustrate this algorithm with a toy example, where the problem is simplified as Fig.10 (a) with $m = 4$, $n = 2$. In the first step, we calculate the penalty cost pc_{row_j} for each row and pc_{col_k} for each column of the tableau in Fig.10 (a). Penalty cost is determined by subtracting the lowest unit cost in the row (column) from the next lowest unit cost. The penalty costs of the respective rows and columns have been marked in red color for clarity in Fig.10 (b). Since the third row has the largest penalty cost (pc_{row_3} =11) and $C_{3,1}$ is the lowest unit cost of that row, $Silo_3$ is allocated to $Mill_1$, i.e. $A_{3,:} = [1, 0]$ in WGM or $L_{3,:} = [1, 0]$ in LGM. Then the corresponding row should be crossed out and the demand of $Mill_1$ should minus one, if this results in a zero demand, the first column will be crossed out as well. After adjusting penalty cost for each row and column, the tableau becomes Fig.10 (c), where the changed values are marked in orange. The described procedure will be looped until no rows remained.

Considering the calculation of pc_{col_k} is much more time-consuming compared with pc_{row_j} because $m \gg n$ in WGM and LGM, we modify VAM by only seeking lowest penalty cost among $\{pc_{row_1}, ..., pc_{row_m}\}$. We find this modification makes VAM more efficient while keeps the superiority of the solution. It is because the mechanics of VAM makes it meaningful to take pc_{col_k} into account only when the demand of $Mill_k$ is one, which rarely happens. Thus, we adopt this modification to promote efficiency in this paper.
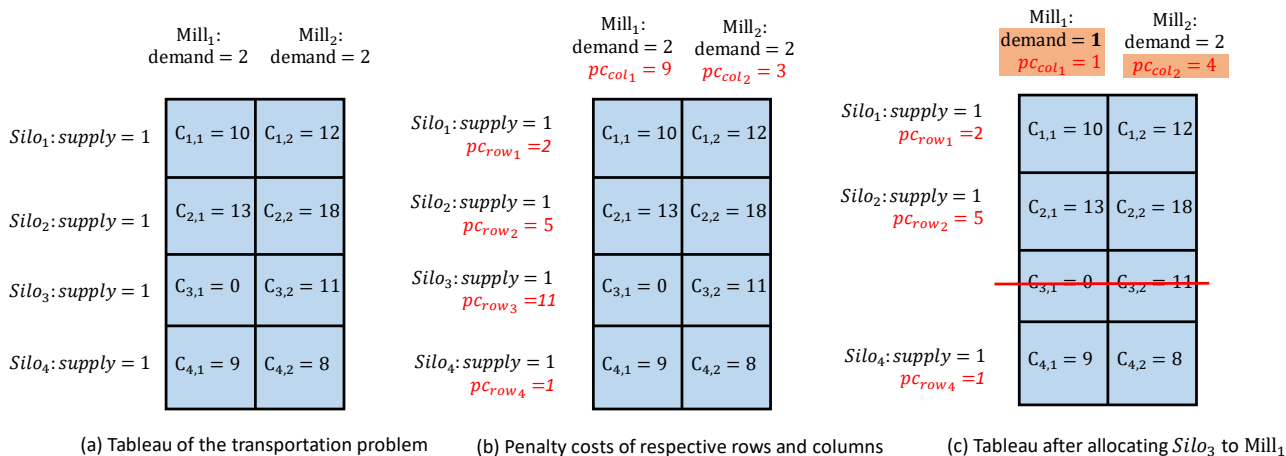

Figure 10. The Vogel approximation method.

A.2. A Strategy for Task Decomposition

To fulfill task decomposition, we introduce a new module to delegator, named subtask selector as shown in Fig.11. The subtask selector is used to allocate the input samples into different subtasks. The expert selector outputs sixteen probabilities, which are partitioned into four groups as well. For each subtask, only one group of probabilities is visible. The experts within each subtask and the corresponding weights of the expert selector are jointly optimized. As for the feature extractor, task predictor, and subtask selector, their weights directly derive from the delegator trained with the setting of four experts and then fixed. During this procedure, the weights of subtask selector derive from the expert selector.
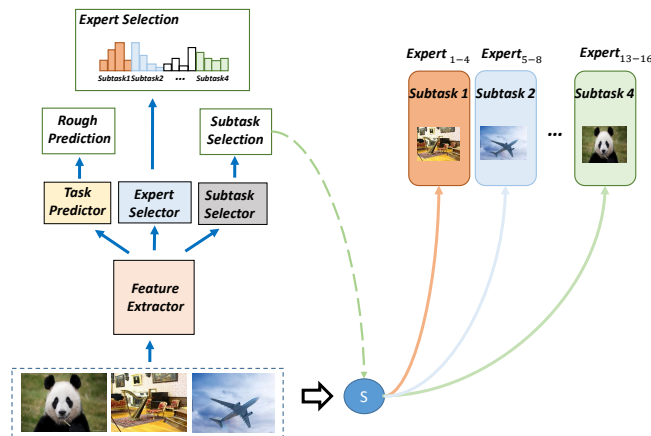
Figure 11. The modified architecture of delegator.

B. Extra details for Experiments

B.1. Experiment details for the ImageNet classification task

To combine with CondConv, we replace the convolutions within each inverted residual block of the experts with CondConv ($expert\_num = 4$). To take advantage of PWLU, we replace all activation layers except those that have tiny input feature maps as illustrated in Zhou et al. (2021). Models are trained using SGD optimizer with 0.9 momentum. We use a mini-batch size of 4096, and a weight decay of 0.00002. Cosine learning rate decay is adopted and the number of training iterations is 313000. We use the augment policy searched by Cubuk et al. (2019) as well (fixed auto-augment). Similar as BasisNet, we use knowledge distillation with EfficientNet-B2 (Tan & Le, 2019; Xie et al., 2020) as the teacher. The learning rate is 0.8/1.6 for CoE-Small/Large and dropout rate is 0.2. The stochastic depth (Huang et al., 2016) is used except for TinyNet-E with a survival probability of 0.8. **We think only when the overfitting problem is solved can task accuracy reflect model capacity exactly. Because this paper is concerned with improving the model capacity with limited computation cost, we use knowledge distillation, fixed auto-augment and stochastic depth to overcome the overfitting problem. Nonetheless, we also conduct ablations for them as shown in Appendix B.2.1**

B.2. Ablation Studies for CoE

B.2.1. ABLATION STUDY FOR THE TRAINING TRICKS

Knowledge distillation (KD), auto-augment (AA) and stochastic depth (SD) are widely-used strategies to overcome the overfitting problem. We think only when the overfitting problem is solved can task accuracy reflect model capacity exactly. Because this paper is concerned with improving the model capacity with limited computation cost, we use these strategies. Nonetheless, we conduct ablations for them in this section. We adopt the CoE-Large setting with 4 experts. Results are shown in Table 8. We find KD extremely important for CoE, it may indicate CoE is easy to be overfitted. In addition, SD decreases the accuracy of CoE. By removing SD, CoE-Large (4 experts) boosts the accuracy from 79.9% to 80.2%. Perhaps, it is because SD makes the capacity of delegator and each expert too tiny (Gontijo-Lopes et al., 2021).

B.2.2. EFFECT OF EXPERT NUMBER

We analyze the number of experts in this section, including 1, 4, and 16 experts. The results are shown in Table 9. Using one expert brings little improvement compared with the original model. When increasing the number of experts, the accuracy becomes 1.9% better with four experts and 2.9% better with sixteen experts. It demonstrates CoE can make full use of multiple experts, leading to a large collaborative benefit. What's more, the accuracy also reaches 79.9% by combining CondConv with OFA-230. In this manner, CoE can further enhance the accuracy to 80.8/81.5% with 4/16 experts.

Table 8. Ablation study for the training tricks. "KD", "AA" and "SD" denotes knowledge distillation, auto-augment and stochastic depth respectively.

KD	AA	SD	Experts	FLOPs	TOP-1 Acc
√	√	√	4	220M	**79.9%**
√	√		4	220M	**80.2%**
√		√	4	220M	79.4%
	√	√	4	220M	76.2%
√			4	220M	79.7%
	√		4	220M	**76.3%**
		√	4	220M	75.2%
			4	220M	75.1%

Table 9. Comparison among different number of experts. "CC" indicates CondConv.

Method	Experts	FLOPs	Acc.
OFA-230	-	230M	78.0%
CoE-Large	1	220M	78.0%
	4	220M	79.9%
	16	220M	80.9%
CC-OFA-230	-	242M	79.9%
CoE-Large + CC	1	214M	79.9%
	4	214M	80.8%
	16	214M	81.5%

B.2.3. EFFECT OF EARLY TERMINATION

The original OFA-230 has 78.0% top-1 accuracy with 230M FLOPs. We can introduce a MobileNetV3-Small to conduct early termination. By varying the threshold, we get a series of accuracies and FLOPs as shown in Fig.12. It can seen that the accuracy becomes 78.0% with 220M FLOPs. This indicates the computation cost brought by MobileNetV3-Small is eliminated via early termination strategy. Inspired by this, we expect to eliminate the computation cost brought by delegator via early termination as well. It does reduce the computation cost by 60/66M FLOPs, demonstrating the effectiveness of early termination.

B.3. Re-evaluation with Reassessed Labels

As described in paper (Beyer et al., 2020), the validation set labels have a set of deficiencies that makes the recent progress on ImageNet classification benchmark suffer from overfitting to the artifacts. To verify the generalization, we use the Reassessed Labels (ReaL) (Beyer et al., 2020) to re-evaluate our method. The results are shown in Table 10. It can be seen that our method still has a remarkable performance, achieving higher accuracy than the compared methods with significantly smaller FLOPs.

Table 10. ReaL and original top-1 accuracy. "CC" means CondConv.

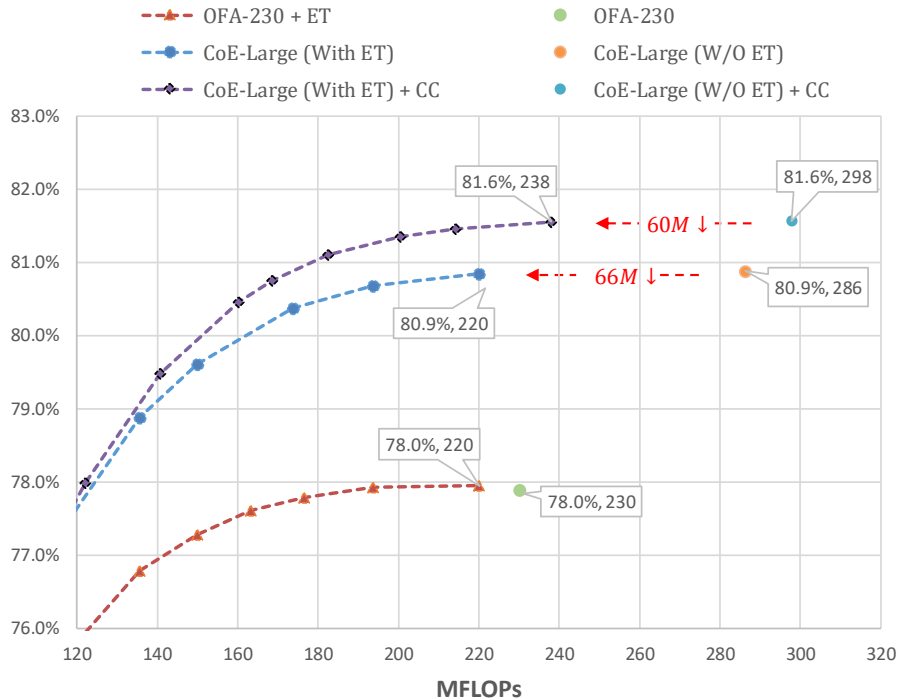| Method	FLOPs	ReaL Acc.	Ori. Acc.
OFA-595 (Cai et al., 2020)	595M	86.0%	80.0%
S4L MOAM (Zhai et al., 2019)	4B	86.6%	80.3%
ResNeXt-101 (Xie et al., 2017)	16B	85.2%	79.2%
ResNet-152 (He et al., 2016)	11B	84.8%	78.2%
CoE-Large	194M	86.5%	80.7%
CoE-Large + CC	214M	86.9%	81.5%

Figure 12. Accuracy v.s. FLOPs. "ET" means Early Termination and "CC" indicates CondConv.

B.4. Experiment details for the translation task

Following (Vaswani et al., 2017; Ott et al., 2019), CoE-Transformer is trained on the standard WMT 2014 English-German dataset, which has a shared source-target vocabulary of about 37000 tokens. As mentioned, an extra token will be added to this vocabulary. The default training setting is identical with the one described in (Vaswani et al., 2017), except for the batch size and learning rate becoming larger following (Ott et al., 2019). Moreover, the parameter α in Eq.5 grows linearly from 0.1 to 0.4 with the training going on. We report BLEU on news2014 with a beam width of 4 and length penalty of 0.6 based on a single model obtained by averaging the last 5 checkpoints following (Vaswani et al., 2017; Ott et al., 2019).

B.5. Analysis of the Learned Expert Selection Patterns

B.5.1. EXPERT SELECTION PATTERNS WHEN EXPERTS SHARE THE SAME ARCHITECTURE

We have analyzed the selection patterns when experts have different architectures, here we focus on the case that all experts share the same architecture. We adopt the CoE-Large setting with four experts.

Considering many works (Yan et al., 2015; Mullapudi et al., 2018) select branches based on the category, we firstly experiment to observe the relationship between selection patterns and rough prediction of the delegator on ImageNet validation set. Based on the predicted class of rough prediction, the validation set can be partitioned into 1000 subsets. Then we calculate the probabilities to select each expert within each subset and get 1000 probability vectors. After clustering, we plot the probability vectors on Fig.13, where each column represents a probability vector. It can be seen that samples with the same rough prediction class are assigned to different experts. Therefore, we can conclude that the expert is not always selected based on category.

Besides, we further make qualitative analysis on the ImageNet validation set and find some interesting patterns. For example, we find that images predicted as "meat market" are most likely to be assigned to the fourth expert if humans are contained. We show those images in Fig.14. It can be seen, 27 images are assigned to the fourth expert, among which 22 images contain humans with a ratio of 81.5%. By contrast, among the 32 images assigned to other experts, only 7 images contain humans with a ratio of 21.9%. This indicates CoE learns the expert selection patterns automatically, it can be based on properties other than the category.
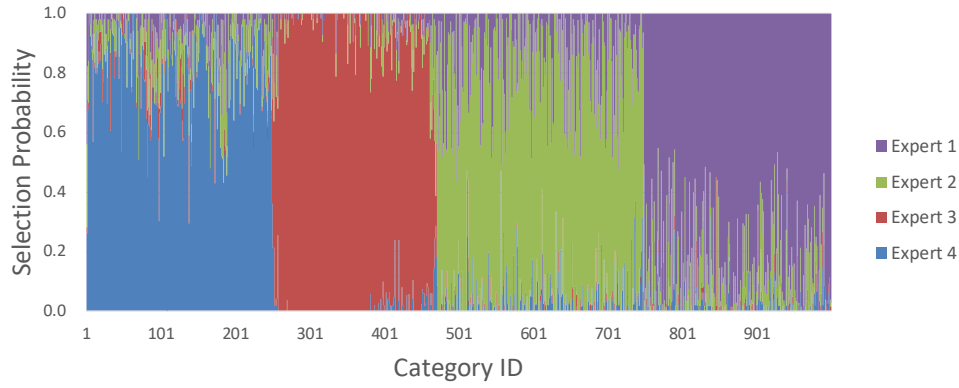
Figure 13. Selection probabilities for each expert. The horizontal axis indicates the rough prediction class. There are a total of 1000 columns, each one represents a probability vector for selecting experts. Probability vectors are clustered for better visualization.

Samples assigned to the **first** expert:

Samples assigned to the **second** expert:

Samples assigned to the **third** expert:

Samples assigned to the **fourth** expert:

: Humans are contained.

: Humans are not contained.

Figure 14. Images that are predicted as 'meat market' by the delegator. They are partitioned into four groups based on which expert is selected. The red border indicates humans are contained, green border indicates humans are not contained.