COLLABORATION OF EXPERTS: ACHIEVING 80% TOP-1 ACCURACY ON IMAGENET WITH 100M FLOPS

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we propose a Collaboration of Experts (CoE) framework to pool together the expertise of multiple networks towards a common aim. Each expert is an individual network with expertise on a unique portion of the dataset, which enhances the collective capacity. Given a sample, an expert is selected by the delegator, which simultaneously outputs a rough prediction to support early termination. To make each model in CoE play its role, we propose a novel training algorithm that consists of three components: weight generation module (WGM), label generation module (LGM) and selection reweighting module (SRM). Our method achieves the state-of-the-art performance on ImageNet, 80.7% top-1 accuracy with 194M FLOPs. Combined with PWLU activation function and CondConv, CoE further achieves the accuracy of 80.0% with only 100M FLOPs for the first time. More importantly, CoE is hardware-friendly, achieving a 3∼6x speedup compared with some existing conditional computation approaches. Experimental results on translation task also show the strong generalizability of CoE.

1 INTRODUCTION

From small systems like organic cells to complicated human society, the accomplishment of various tasks relies on the collaboration of multiple individuals. It can pool together the skills and creativity of each individual towards a common aim, increasing the chance of success. This is a consensus since long ago: as early as 1624, John Donne has popularised the phrase "No man is an island". Similarly, in machine learning society, researchers have also studied the collaboration of different models for decades. Just like human beings, different models naturally have diverse properties. For example, neural networks with more layers in early stages can extract low-level feature more delicately, otherwise, high-level semantic feature is improved Zhang et al. (2020c). It is still the truth even if models share the same architecture because the random seed yields a great deal of randomness for the optimization Hansen & Salamon (1990). Therefore, the outcome on a given task should be improved if the expertise of multiple models is combined.

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 Zhang et al. (2021); Cheng et al. (2020); Yan et al. (2015); Shazeer et al. (2017) alleviates this issue via delegation scheme, i.e. assigning one or several, rather than all models, conditionally to make the prediction. Some recently proposed conditional computation methods Zhang et al. (2020b); Yang et al. (2019); Zhang et al. (2021) have achieved remarkable performance with low computation cost based on dynamic convolution. Nonetheless, they usually have high memory access cost (MAC) and a low degree of parallelism, which increases 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. Each expert is an individual model with expertise on a unique portion of the dataset. The delegator will give a rough prediction and make the expert selection simultaneously. Only when the rough prediction is unreliable Huang & Chen (2018); Teerapittayanon et al. (2016), the expert selected by delegator makes the refined prediction. Moreover, we only need to load the selected expert into memory, thus keep the ratio

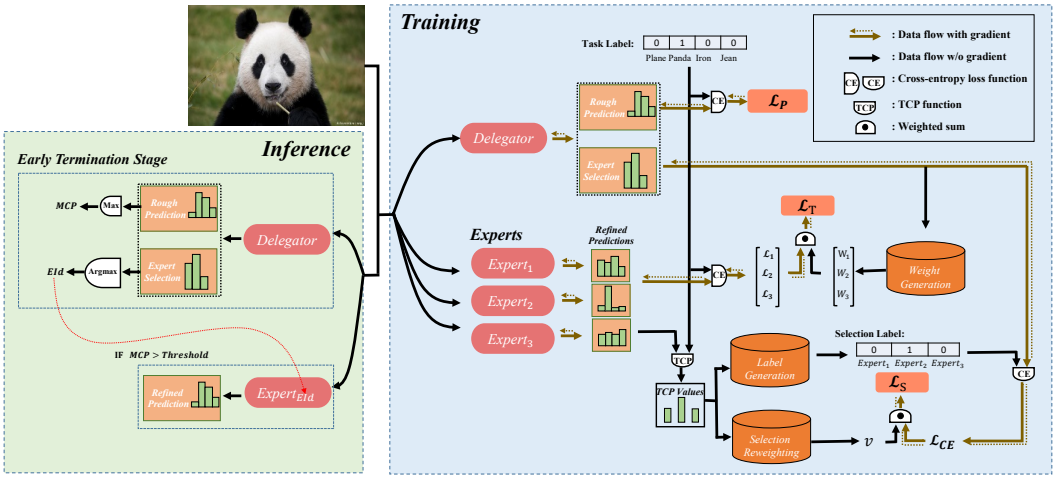


Figure 1: Framework of CoE with 3 experts. The delegator outputs the probabilities to select each expert and a rough prediction to support early termination. Each input is processed through all experts during training, but only through at most one expert during inference.

of MAC to FLOPs as a constant. By contrast, dynamic convolution methods Zhang et al. (2020b; 2021) need to load a large number of parameters, namely basis models or experts, to synthesize the input-dependent ones. It enlarges the 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 three components: weight generation module (WGM), label generation module (LGM) and selection reweighting module (SRM). To make each expert focus on the unique portion of dataset assigned to itself, we propose WGM to reweight losses of them. LGM is used to generate the selection label which indicates the suitable expert. After that, the delegator is optimized based on this label with weighted cross-entropy loss $\mathcal{L}_S = v \cdot \mathcal{L}_{CE}$, where v is obtained from SRM and $\mathcal{L}_{CE}$ is cross-entropy loss. If there exist multiple suitable experts for the given input, there is no need to force delegator to select the exact expert indicated by selection label, thus SRM makes the value of v smaller. This drives the delegator to focus on samples whose recognition results are sensitive to expert selection.

Experimental results on ImageNet classification task Deng et al. (2009) demonstrate the superiority of our method. CoE achieves 78.2/80.7% top-1 accuracy with only 100/194M FLOPs, while the ensembled models Hansen & Salamon (1990) can only achieve 79.6% accuracy with 920M FLOPs. Compared with dynamic network approaches Yang et al. (2019); Zhang et al. (2021); Cheng et al. (2020), CoE is more hardware-friendly. It not only outperforms the state-of-the-art 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 Zhang et al. (2020b); Yang et al. (2019) and further improve the accuracy to 79.2/81.5% with 102/214M FLOPs. More surprisingly, we achieve an accuracy of 80.0% with only 100M FLOPs for the first time by using PWLU activation function Zhou et al. (2021). Experimental results on translation task also show 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) and demonstrate that it can lead to outstanding performance with little computation cost. More importantly, this framework is hardware-friendly and can achieve real speedup.
- We present a novel training algorithm and optimization strategy for CoE. The core insight is to promote diversity within experts by distributing their expertise over different portions of the dataset.
- We update the state-of-the-art on ImageNet for mobile setting, achieving 80.0% top-1 accuracy with only 100M FLOPs for the first time.

2 RELATED WORK

2.1 ENSEMBLE LEARNING

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, our method 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 Zhong et al. (2018; 2021); Fang et al. (2019); Zhang et al. (2020a); Cai et al. (2020). It is because these models are compact enough and have few redundant parameters. By contrast, our method is free from the compactness of experts and compatible with various models.

2.2 DYNAMIC NETWORKS

Dynamic network methods can achieve high performance with low computation cost by conditionally varying the network parameters Zhang et al. (2020b); Yang et al. (2019); Zhang et al. (2021); Chen et al. (2020) or network architectures Cheng et al. (2020); 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 our method 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) enable the direct training of Router by scaling the output feature of experts with the predicted gate-values of Router. These methods aim at conditionally selecting a specific layer or block. Differently, our method can take more advantage of conditional computation as the selection of whole network makes every parameter input-dependent. The recently proposed Dynamic Convolution methods Zhang et al. (2020b); Yang et al. (2019); Chen et al. (2020) share the same 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 memory access cost and low degree of parallelism Ma et al. (2018). By contrast, our method 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 Zhang et al. (2020b); Yang et al. (2019); Zhang et al. (2021) or architectures Cheng et al. (2020); Yuan et al. (2020), these methods cannot process samples in batch. Differently, our method supports batch processing because the number of experts is limited and each one of them corresponds to many test samples.

3 METHOD

In this paper, we propose the CoE framework which pools together the expertise of multiple networks towards a common aim. As shown in Fig. 1, the framework includes a delegator and n experts, a total of $n+1$ individual neural networks. To make the best of limited capacity, each expert is encouraged to focus on just one unique portion of the dataset. This is achieved by WGM which reweights training losses of experts. Meanwhile, the expert is selected by delegator. It is trained with cross-entropy loss and selection label generated from LGM. Moreover, we propose SRM to make the delegator focus on samples whose recognition result is sensitive to expert selection. To take advantage of early termination Huang & Chen (2018); Teerapittayanon et al. (2016) during inference, the delegator also outputs a rough prediction. Specifically, we accept the rough prediction if its *Maximum Class Probability* (probability of the predicted class) is larger than threshold, otherwise, adopt the selected expert for a refined prediction. Since the inference of delegator is conducted all the time, we prefer to make it more lightweight than expert. It consists of three modules: feature extractor, task predictor and expert selector as shown in Fig. 2. Based on the feature derived from feature extractor,

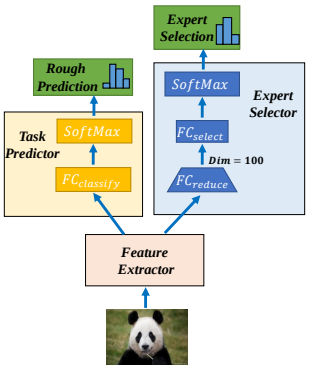


Figure 2: Architecture of delegator.

task predictor and expert selector output the probabilities for classification and expert selection respectively. We describe the training strategies comprehensively in the following subsections. The number of samples and experts are denoted as m and n respectively throughout this section.

3.1 WEIGHT GENERATION MODULE (WGM)

Our proposed method achieves remarkable performance by the collaboration of multiple networks. To maximize the collective capacity, we encourage each expert to focus on a unique portion of the dataset. To accomplish this goal, we propose WGM to reweight losses of different experts.

Given m samples, the delegator outputs a probability matrix $\boldsymbol{P}_{m\times n} \in \boldsymbol{R}^{m\times n}$, whose element $P_{j,k} \in [0,1]$ represents the probability of assigning the j-th sample to the k-th expert. WGM takes $\boldsymbol{P}_{m\times n}$ as input and outputs a weight matrix $\boldsymbol{W}_{m\times n}$ by smoothing and normalizing an assignment matrix $\boldsymbol{A}_{m\times n}$, which is a binary matrix with one-hot row vectors. $A_{j,k} = 1$ means the j-th sample is assigned to the k-th expert during training, thus $W_{j,k}$ should be larger. Given that the main concern of this paper is not the optimization of network architecture, we suppose no expert is superior to others (***No Superiority Assumption***). For example, even though a larger model can have a better performance, its FLOPs is inevitably larger. Thus, it isn't superior to smaller models in regards to FLOPs. Based on this assumption, the number of samples assigned to each expert should be same, i.e. the assignment matrix $\boldsymbol{A}_{m\times n}$ needs to satisfy $\sum_j A_{j,k} = m/n$. Thus, $\boldsymbol{A}_{m\times n}$ can be determined by solving the following problem:

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

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. $-P_{j,k}$ 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)), which is a short-cut approach to solve BTP and invariably obtains a very good solution. It is introduced in appendix A.1.

At the early stage of training, neither delegator nor experts are trained well. Thus we cannot trust the assignment matrix $\boldsymbol{A}_{m\times n}$ and need to make the loss weight for selected and unselected experts similar. We achieve this by smoothing $\boldsymbol{A}_{m\times n}$ to $\overline{\boldsymbol{A}}_{m\times n}$ with Equation. 2, where α grows linearly from 0.2 to 0.8 with the training going on,

$$\overline{A}_{j,k} = \begin{cases} \alpha + \frac{1-\alpha}{n}, & if\ A_{j,k} = 1 \\ \frac{1-\alpha}{n}, & if\ A_{j,k} = 0 \end{cases} . \tag{2}$$

Finally, the output of WGM (i.e. $\boldsymbol{W}_{m\times n}$) is obtained by normalizing $\overline{\boldsymbol{A}}_{m\times n}$ with the coefficient $\mathcal{Z} = \sum_j \overline{A}_{j,k} = \frac{m}{n}$:

$$W_{j,k} = \frac{\overline{A}_{j,k}}{\mathcal{Z}}. \tag{3}$$

3.2 LABEL GENERATION MODULE (LGM)

LGM generates the selection label to supervise delegator, aiming at maximizing the overall accuracy of CoE. Selection label is a one-hot vector that indicates the suitable expert for a given input. To find the suitable expert, we leverage *True Class Probability* (TCP, Corbière et al. (2019)) to measure the suitability of Expert_k:

$$\text{TCP}_{j,k} = P(Y = y_j | x_j, \text{Expert}_k), \tag{4}$$

where, x_j is the j-th sample, Y and y_j are the predicted and true class. According to *No Superiority Assumption*, the suitability for each expert should satisfy identical distribution over the dataset. Thus we refine the criterion for suitability from $\text{TCP}_{j,k}$ to $S_{j,k}$ via standardization:

$$S_{j,k} = \frac{\text{TCP}_{j,k} - Mean(\text{TCP}_{:,k})}{Std(\text{TCP}_{:,k})}, \tag{5}$$

where, Mean($\text{TCP}_{:,k}$) and Std($\text{TCP}_{:,k}$) are mean value and standard deviation of TCPs for Expert_k. We denote the output of LGM as a binary matrix $\boldsymbol{L}_{m\times n}$, each row of which is a selection label. To maximize the total suitability, $L_{m\times n}$ is determined by solving the following problem:

$$\begin{aligned} \min \quad & \sum_{j,k} -S_{j,k} * L_{j,k} \\ s.t. \quad & L_{j,k} \in \{0,1\}, \quad \sum_k L_{j,k} = 1, \quad \sum_j L_{j,k} = \frac{m}{n} \end{aligned} \tag{6}$$

This problem can also be modeled as BTP, and solved via VAM as described in section 3.1.

3.3 Selection Reweighting Module (SRM)

Cross-entropy loss $\mathcal{L}_{CE}$ can force delegator to select the exact expert indicated by selection label generated from LGM. However, there may exist multiple suitable experts for some samples. In this case, it only causes little to no drop in accuracy if the input is assigned to other experts, namely the recognition result is insensitive to expert selection. As the capacity of delegator is limited, less attention is needed for those samples. Considering this, we propose SRM to dynamically adjust the weight of $\mathcal{L}_{CE}$.

The output of SRM is denoted as $\{v_j | j = 1, ..., m\}$, which are weights for $\mathcal{L}_{CE}$ over m samples. SRM firstly calculates the suitability matrix $\boldsymbol{S}_{m\times n}$ via Equation. 4, 5. Then the standard deviation Std($S_{j,:}$) for row vector $S_{j,:}$ is calculated. A small Std($S_{j,:}$) indicates the experts have similar suitability, thus v_j is smaller. Therefore, SRM determines the value of v_j by:

$$v_j = \frac{Std(S_{j,:})}{\sum_i Std(S_{i,:})}. \tag{7}$$

3.4 Training method

The training procedure of CoE is composed of two stages. During the first stage, only feature extractor and task predictor of delegator are trained to minimize the regular cross-entropy loss $\mathcal{L}_P$. After that, we fix these two modules, jointly train the expert selector and experts with the following loss:

$$\mathcal{L}_{Total} = \eta * \mathcal{L}_S + \mathcal{L}_T, \tag{8}$$

here, the hyperparameter η is set as 0.8.

$\mathcal{L}_S$ is used to optimize the expert selector. Based on the selection label $L_{j,:}$ obtained from LGM, we can get the cross-entropy loss $\mathcal{L}_{CE}^j$ for the j-th sample. Then $\mathcal{L}_S$ is determined by the weighted sum of $\{\mathcal{L}_{CE}^j | j = 1, \ldots, m\}$ with weights $\{v_j | j = 1, \ldots, m\}$ output by SRM:

$$\mathcal{L}_S = \sum_j v_j * \mathcal{L}_{CE}^j. \tag{9}$$

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

$$\mathcal{L}_T = \sum_{j,k} W_{j,k} * \mathcal{L}_{j,k}. \tag{10}$$

Each of our experiments includes either four or sixteen experts in this paper. When using four experts, the training method is same as described. However, we will meet slow-convergence problem if the number of experts is sixteen. It is because most of the samples, specifically $\frac{15m}{16}$ samples, have tiny weights for one expert according to WGM. To alleviate this problem, we propose a strategy which is described in Appendix A.2.

4 Experiments

We conduct the main experiments on ImageNet classification task Deng et al. (2009). After comparing with some popular methods in terms of accuracy, computation cost and inference latency, we analyze the effect of expert number and early termination respectively. Then, we discuss the training cost of CoE and compare it with some existing model collaboration methods. Moreover, the generalization of CoE is verified. Finally, we try to analyze the reasonability of learned expert selection pattern. Experiments are conducted 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 the CoE-Large, MobileNetV3-Small Howard et al. (2019) with 56M FLOPs is adopted to construct the delegator by analogy. We use OFA-230 as the experts. We have also tried to introduce CondConv Zhang et al. (2020b) and PWLU activation fuction Zhou et al. (2021) to achieve the extreme performance. 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 fixed auto-augment Cubuk et al. (2019) as well. Inspired by BasisNet Zhang et al. (2021), we use knowledge distillation Hinton et al. (2015) 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.

4.2 Results and Analysis

4.2.1 Accuracy and Computation Cost

We compare with some efficient networks in Table.1 and Fig. 3, where statistics on referenced baselines are directly cited from original papers. Our method achieves 78.2% and 80.7% top-1 accuracy with 16 experts and averaged FLOPs as 100M and 194M respectively. Compared with OFA Cai et al. (2020), our method reduces the FLOPs from 230M to 100M and from 595M to 194M respectively, with better top-1 accuracy. Compared with EfficientNet-B1 with noisy student training Xie et al. (2020), our method also reduces the FLOPs by 3.6x while improving the accuracy by 0.5%. Though dynamic networks like GFNet Wang et al. (2020), CondConv Yang et al. (2019) and BasisNet Zhang et al. (2021) are more efficient than traditional networks, our method still has significantly higher accuracy with smaller FLOPs. Compared with these approaches, our method improves the accuracy by 2.2/2.4/0.7% respectively. When combined with CondConv, we can achieve 79.2% and 81.5% top-1 accuracy with only 102M and 214M FLOPs respectively, which indicates our method 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 top-1 accuracy of only 80.5%. More surprisingly, we achieve the accuracy of 80.0% with only 100M FLOPs for the first time by further making use of PWLU.

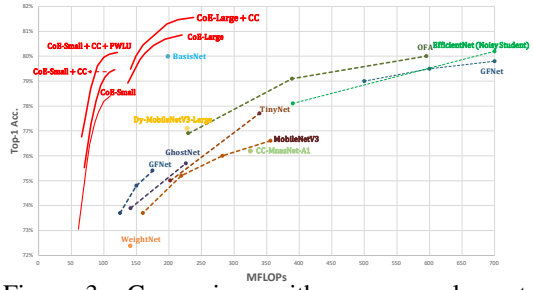


Figure 3: Comparison with some popular networks on ImageNet. "CC" and "PWLU" means CondConv and PWLU activation function.

4.2.2 Inference Speed on Hardware

Compared with conditional computation methods Yang et al. (2019); Zhang et al. (2021), CoE is more hardware friendly. To verify the advantage, we also 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. When the mini-batch size is larger than one, our method firstly uses delegator to partition ImageNet validation set into $n+1$ groups, where one group is classified based on the rough prediction and other groups rely on the refined prediction of corresponding expert.

We report the averaged latency on the ImageNet validation set in Table. 2. From the table, 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

Table 1: CoE performance on ImageNet. "CC" and "KD" indicates conditional computation approach and knowledge distillation strategy.

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

number of parameters for this synthesis, which brings a large memory access cost. This is why CoE can reduce 14.09% latency when the mini-batch size is one. Finally yet importantly, BasisNet and CondConv do not support batch processing, while CoE can take advantage of batch processing to further achieve a 3.1/6.1x speedup compared with BasisNet/CondConv-EfficientNet-B0.

Table 2: The accuracy and CPU latency for different methods.

Models	CPU Latency/Instance (ms)		FLOPs	Accuracy
	Batchsize=1	Batchsize=64		
MobileNetV3-Small Howard et al. (2019)	14.77	4.18	56M	67.4%
GhostNet 1.0x Han et al. (2020a)	39.91	16.50	141M	73.9%
TinyNet-B Han et al. (2020b)	34.58	19.44	202M	75.0%
GhostNet 1.3x Han et al. (2020a)	43.94	29.70	226M	75.7%
OFA-230 Cai et al. (2020)	33.52	15.21	230M	76.9%
EfficientNet-B0 Yang et al. (2019)	49.12	35.21	391M	77.2%
TinyNet-A Han et al. (2020b)	45.76	23.71	339M	77.7%
CondConv-EfficientNet-B0 Yang et al. (2019)	81.81	-	413M	78.3%
BasisNet Zhang et al. (2021)	40.61	-	198M	80.0%
CoE-Large	34.89	13.30	194M	80.7%

4.2.3 Effect of Expert Number and Early Termination

The effect of experts number are shown in Table.3. Using one expert brings no 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. The effect of early termination are shown in Fig. 4a, where MobileNetV3-Smalll is adopted to conduct early termination. It reduces the computation cost of CoE by 60/66M FLOPs, demonstrating the effectiveness of early termination.

4.2.4 Analysis of the training cost

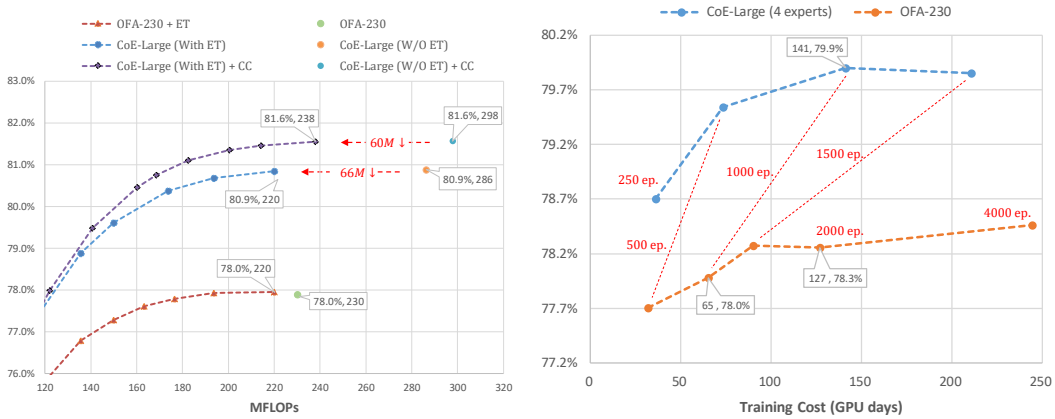
To achieve superior performance with less inference FLOPs and latency, CoE may consume more training time. For example, the applying of CoE (4 experts) on OFA-230 improves the accuracy from 78.0% to 79.9%, but at the expense of a 2.2x training cost. To verify whether the improvement still exists with similar training cost, we get a series of accuracies by varying the number of training epochs as shown in Fig. 4b, where "xx ep." means the number of training epochs is "xx" and 32 Tesla-V100-PCIe-16GB GPUs are used for training. It is seen that CoE can boost the performance (from 78.3% to 79.9%) even if the training cost is controlled to be similiar (127 vs 141 GPU days).

Table 3: 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%

Table 4: 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	220M	79.9%
	16 Experts	220M	80.9%



(a) Accuracy v.s. FLOPs. "ET" means Early Termination and "CC" indicates CondConv.

(b) Accuracy v.s. training cost for OFA-230 and CoE-Large (with 4 OFA-230 experts).

Figure 4: Accuracy v.s. FLOPs and training cost on ImageNet.

4.3 Comparison with Other Model Collaboration Approaches

4.3.1 Comparison with Regular Model Ensemble

We train four OFA-230 models with different initialization seeds. The results are shown in Table. 4. The random seed only causes minor variety in accuracy. However, an improvement of 1.6% is still achieved via ensemble. It is because these models fall into different local minima, yielding the diversity of output. Compared with the ensemble, CoE achieves 0.3/1.3% higher accuracies. It indicates CoE can realize more potential of each individual model. Besides, CoE keeps the computation cost constant, while model ensemble increases the computation cost by four times.

4.3.2 Comparison with Model Selection Method

HD-CNN Yan et al. (2015) and HydraNets Mullapudi et al. (2018) select branches based on the category. MoE Shazeer et al. (2017) and Switch Transformer Fedus et al. (2021) enable the direct training of Router by scaling the output feature of experts with gate-values predicted from Router. Despite these methods are originally designed to conditionally select a specific layer or block, we apply them to the expert selection.

To select the expert based on category, the categories should be partitioned into n groups, where n is the number of experts. We try two schemes: random or clustering-based partition Mullapudi et al. (2018). Then, an 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 $\boldsymbol{A}_{m\times n}$ with Equation 2 and 3. $\boldsymbol{A}_{m\times n}$ is obtained directly based on the rough prediction. We can enable the direct training of expert selector as well. It is achieved by scaling the loss of each expert, rather than the output feature of each branch as the original paper Shazeer et al. (2017); Fedus et al. (2021). While CoE optimizes the expert selector via weighted cross-entropy loss $\mathcal{L}_s$ (Equation 9). The

results with 4 experts are shown in Table 5. It can be seen that CoE outperforms the compared methods, demonstrating a better collaboration pattern is learned.

Table 5: Comparison with other model selection methods.

Method		FLOPs	Top-1 Acc.
Category-Based	Random Partition	220M	78.3%
	Clustering-Based Partition	220M	77.5%
Selector-Based	Direct Training	220M	78.7%
	CE-Based Training (Ours)	220M	79.9%

4.4 Analysis of the Generalization

To verify the generalizability, we conduct two extra experiments: ***generalizing CoE to translation task*** and using Reassessed Labels (ReaL) to ***re-evaluate CoE***. We mainly introduce the first one here, another one are discussed in Appendix B.1.

To generalize CoE to translation task, we build a CoE-Transformer model based on Transformer (base model) Vaswani et al. (2017). Considering the decoding procedure is much more time-consuming than encoding because target tokens are generated one by one, only the decoder of CoE-Transformer is conditionally selected. CoE-Transformer has four decoders, given a sentence, one decoder will be selected to decode the features output by encoder. To select the decoder, an extra constant token is added at the beginning of each sentence, the feature of this token output by encoder is input to the Expert Selector (Figure. 2) for expert (decoder) selection. During training, the TCP of a sentence is obtained by averaging the TCPs of each token.

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 into this vocabulary. We adopt the same training and evaluating setting as Ott et al. (2019), more details are shown in Appendix B.2. Although CoE-Transformer has a larger number of parameters compared with Transformer (base model), the memory access cost (MAC) is nearly identical. We calculate MAC based on the number of parameters loaded given a sentence and report it on Table. 6 as well. From Table. 6 we can see, CoE-Transformer outperforms the regular Transformer (base model) by a large margin and achieves similar performance with Transformer (big).

Table 6: The BLEU scores on newstest2014 (English-to-German).

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

4.5 Reasonability of the Learned Expert Selection Pattern

We also conduct experiments to analyze the expert selection pattern of CoE and find it quite reasonable. When experts have different architectures, the delegator learns to assign easy samples to smaller experts and complex samples to heavier experts. When experts share the same architecture, the delegator can learn the expert selection pattern automatically, which is based on properties other than category (e.g. whether humans are contained). The details are shown in Appendix B.3.

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 our method on both accuracy and real speed. We also analyze the collaboration pattern and find it has 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 can be conducted to solve the problems of lifelong learning by updating experts.

REFERENCES

Lucas Beyer, Olivier J. Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are we done with imagenet? *CoRR*, abs/2006.07159, 2020.

Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. 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.

Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11030–11039, 2020.

An-Chieh Cheng, Chieh Hubert Lin, Da-Cheng Juan, Wei Wei, and Min Sun. Instanas: Instance-aware neural architecture search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3577–3584, 2020.

Charles Corbière, Nicolas THOME, Avner Bar-Hen, Matthieu Cord, and Patrick Pérez. 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.

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

Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Bichen Wu, Zijian He, Zhen Wei, Kan Chen, Yuandong Tian, Matthew Yu, Peter Vajda, et al. Fbnetv3: Joint architecture-recipe search using neural acquisition function. *arXiv preprint arXiv:2006.02049*, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Muyuan Fang, Qiang Wang, and Zhao Zhong. BETANAS: balanced training and selective drop for neural architecture search. *CoRR*, abs/1912.11191, 2019.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

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

Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. 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.

Kai Han, Yunhe Wang, Qiulin Zhang, Wei Zhang, Chunjing Xu, and Tong Zhang. Model rubik's cube: Twisting resolution, depth and width for tinynets. In *NeurIPS*, 2020b. URL https://proceedings.neurips.cc/paper/2020/hash/e069ea4c9c233d36ff9c7f329bc08ff1-Abstract.html.

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

Marton Havasi, Rodolphe Jenatton, Stanislav Fort, Jeremiah Zhe Liu, Jasper Snoek, Balaji Lakshminarayanan, Andrew Mingbo Dai, and Dustin Tran. Training independent subnetworks for robust prediction. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=OGg9XnKxFAH.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1314–1324, 2019.

Gao Huang and Danlu Chen. Multi-scale dense networks for resource efficient image classification. *ICLR 2018*, 2018. URL `https://par.nsf.gov/biblio/10064650`.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pp. 646–661. Springer, 2016.

Changlin Li, Guangrun Wang, Bing Wang, Xiaodan Liang, Zhihui Li, and Xiaojun Chang. Dynamic slimmable network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8607–8617, June 2021.

Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 116–131, 2018.

Ningning Ma, Xiangyu Zhang, Jiawei Huang, and Jian Sun. Weightnet: Revisiting the design space of weight networks. *CoRR*, abs/2007.11823, 2020.

Ravi Teja Mullapudi, William R Mark, Noam Shazeer, and Kayvon Fatahalian. Hydranets: Specialized dynamic architectures for efficient inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 8080–8089, 2018.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

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

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

Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2464–2469. IEEE, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pp. 6000–6010, 2017. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need`.

Yulin Wang, Kangchen Lv, Rui Huang, Shiji Song, Le Yang, and Gao Huang. Glance and focus: a dynamic approach to reducing spatial redundancy in image classification. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (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`.

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

Florian Wenzel, Jasper Snoek, Dustin Tran, and Rodolphe Jenatton. Hyperparameter ensembles for robustness and uncertainty quantification. *arXiv preprint arXiv:2006.13570*, 2020.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 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.

Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Zhicheng Yan, Hao Zhang, Robinson Piramuthu, Vignesh Jagadeesh, Dennis DeCoste, Wei Di, and Yizhou Yu. 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.

Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/f2201f5191c4e92cc5af043eebfd0946-Paper.pdf.

Kun Yuan, Quanquan Li, Dapeng Chen, Aojun Zhou, and Junjie Yan. Dynamic graph: Learning instance-aware connectivity for neural networks. *arXiv preprint arXiv:2010.01097*, 2020.

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

Mingda Zhang, Chun-Te Chu, Andrey Zhmoginov, Andrew Howard, Brendan Jou, Yukun Zhu, Li Zhang, Rebecca Hwa, and Adriana Kovashka. Basisnet: Two-stage model synthesis for efficient inference. *arXiv preprint arXiv:2105.03014*, 2021.

Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial autoaugment. In *International Conference on Learning Representations*, 2020a. URL https://openreview.net/forum?id=ByxdUySKvS.

Yikang Zhang, Jian Zhang, Qiang Wang, and Zhao Zhong. Dynet: Dynamic convolution for accelerating convolutional neural networks. *arXiv preprint arXiv:2004.10694*, 2020b.

Yikang Zhang, Jian Zhang, and Zhao Zhong. Autobss: An efficient algorithm for block stacking style search. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 10259–10270. Curran Associates, Inc., 2020c. URL https://proceedings.neurips.cc/paper/2020/file/747d3443e319a22747fbb873e8b2f9f2-Paper.pdf.

Zhao Zhong, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Practical block-wise neural network architecture generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Zhao Zhong, Zichen Yang, Boyang Deng, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Blockqnn: Efficient block-wise neural network architecture generation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(7):2314–2328, 2021. doi: 10.1109/TPAMI.2020.2969193.

Yucong Zhou, Zezhou Zhu, and Zhao Zhong. 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.5 (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.5 (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.5 (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.5 (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.

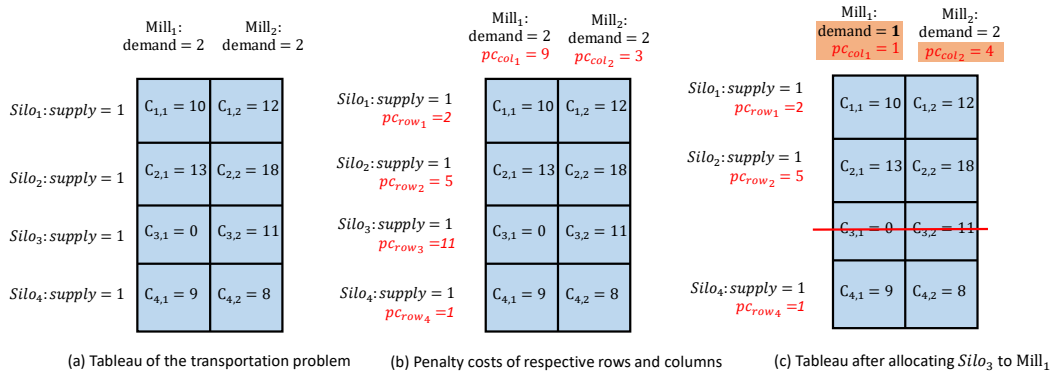


Figure 5: The Vogel approximation method.

A.2 A STRATEGY TO FACE THE LARGE NUMBER OF EXPERTS

Each of our experiments involves either four or sixteen experts. For the four-experts setting, the training method is the same as described. However, we will meet the slow-convergence problem when the number of experts is sixteen. It motivates us to decompose the task into four subtasks, each of which involves four experts and can be trained with the proposed training method. With this strategy, the number of samples assigned to each expert increases from $\frac{m}{16}$ to $\frac{m}{4}$. Because these samples contribute most to the optimization of the expert, the rate of convergence becomes nearly four times faster.

To fulfill task decomposition, we introduce a new module to delegator, named subtask selector as shown in Fig. 6. 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.

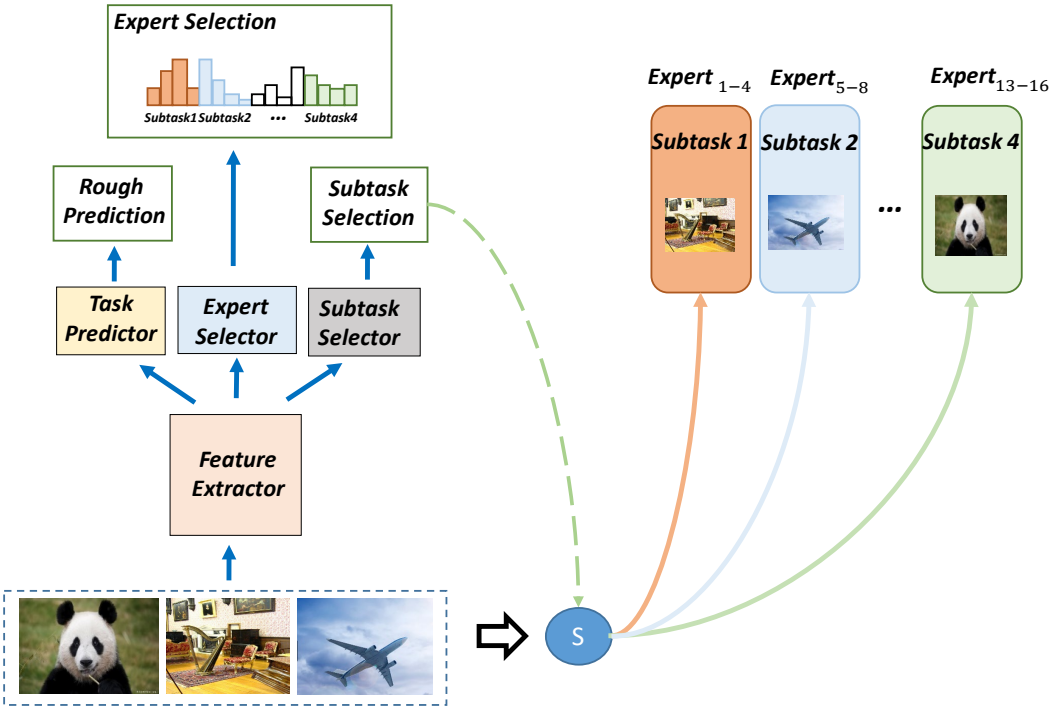


Figure 6: The modified architecture of delegator.

B Extra details for Experiments

B.1 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. 7. It can be seen that our method still has a remarkable performance, achieving higher accuracy than the compared methods with significantly smaller FLOPs.

Table 7: ReaL and original top-1 accuracies. 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%

B.2 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 into 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 become larger following Ott et al. (2019). Moreover, the parameter α in Eqution 2 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.3 REASONABILITY OF LEARNED EXPERT SELECTION PATTERN

B.3.1 EXPERT SELECTION PATTERN WHEN EXPERTS HAVE VARIOUS ARCHITECTURES

Considering TCP can measure the complexity of a given sample, we experiment to analyze the relationship between sample complexity and expert selection. If the experts have different computation cost, it is reasonable to assign easy samples (the ones with large TCP) to smaller experts and complex samples to heavier experts. To verify this, 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. 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. 7, it meets our expectation that the selection probability for smaller models increases with the increase of TCP.

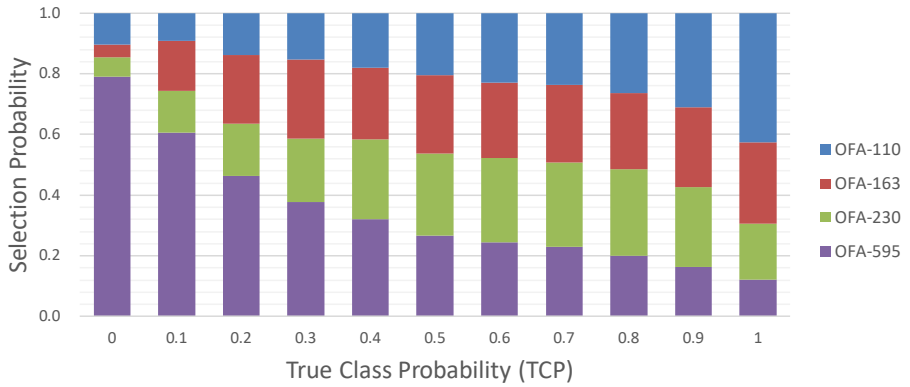


Figure 7: Selection probabilities for each expert at different TCP values.

B.3.2 EXPERT SELECTION PATTERN WHEN EXPERTS SHARE THE SAME ARCHITECTURE

We have analyzed the selection pattern when the 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 pattern and rough prediction of the delegator on ImageNet validation set. We construct each probability vector, whose elements are the selection probabilities for each expert, based on samples of the same rough prediction class. For better visualization, we cluster the 1000 probability vectors and then plot them on Fig. 8. It can be seen that samples with the same rough prediction class are assigned to different experts. Therefore, we can conclude 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. 9. 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 that are assigned to the other experts, only 7 images contain humans with a ratio of 21.9%. This indicates CoE learns the expert selection pattern automatically, it can be based on properties other than the category.

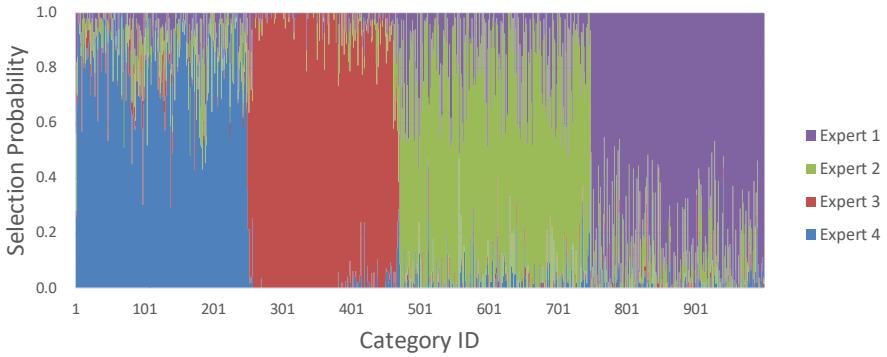


Figure 8: Selection probabilities for each expert. The horizontal axis indicates the rough prediction class. The 1000 probability vectors are clustered for better visualization.

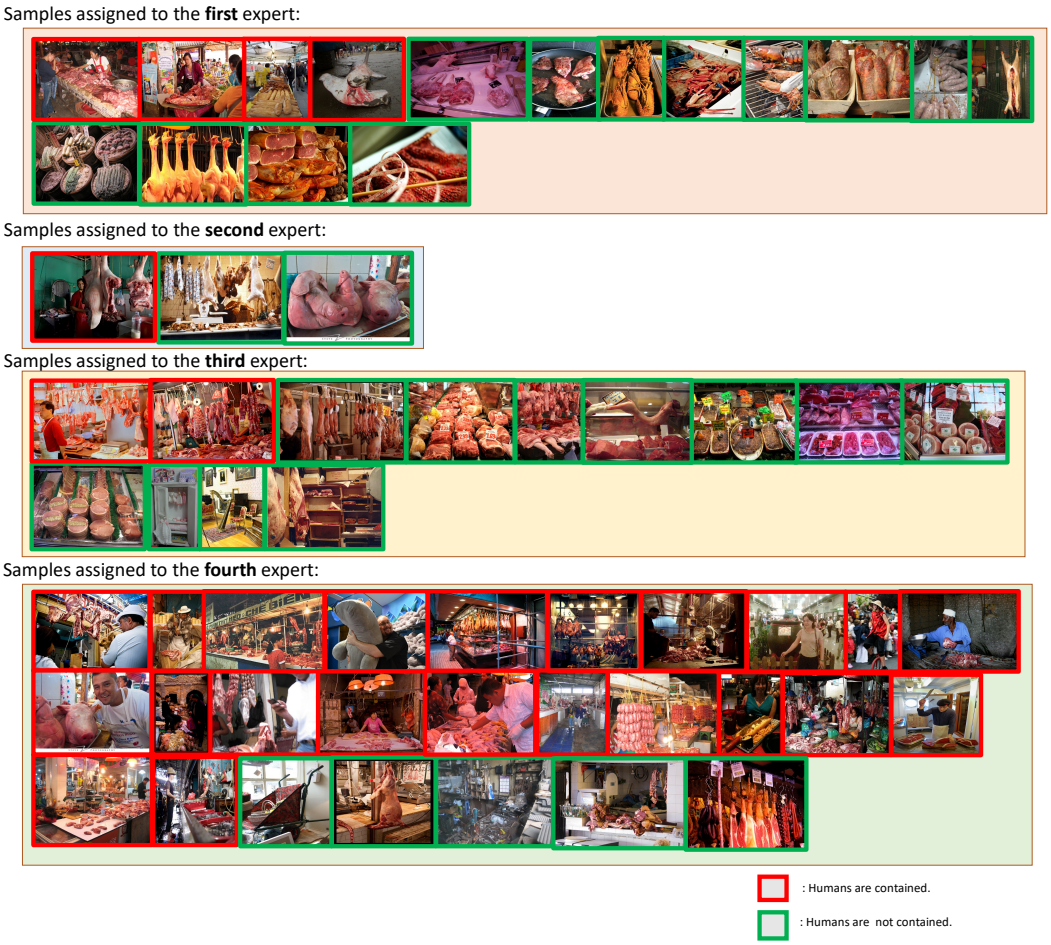


Figure 9: Images that are predicted as 'meat market' by the delegator. They are partitioned into four groups based on which expert is selected. The red border indicates humans are contained, green border indicates humans are not contained.