Revisiting single-gated Mixtures of Experts

Amélie Royer
aroyer@qti.qualcomm.com
Ilia Karmanov
ikarmano@qti.qualcomm.com
Andrii Skliar
askliar@qti.qualcomm.com
Babak Ehteshami Bejnordi
behtesha@qti.qualcomm.com
Tijmen Blankevoort
tijmen@qti.qualcomm.com

Qualcomm AI Research
Science Park 400
Amsterdam
The Netherlands

Abstract

Mixture of Experts (MoE) are rising in popularity as a means to train extremely large-scale models, yet allowing for a reasonable computational cost at inference time. Recent state-of-the-art approaches usually assume a large number of experts, and require training all experts jointly, which often lead to training instabilities such as the router collapsing. In contrast, in this work, we propose to revisit simple single-gate MoE, which allows for more practical training. Key to our work are **(i)** a base model branch acting both as an early-exit and an *ensembling regularization* scheme, **(ii)** a simple and efficient *asynchronous* training pipeline without router collapse issues, and finally **(iii)** an automatic *per-sample* clustering-based initialization. We show experimentally that the proposed model obtains efficiency-to-accuracy trade-offs comparable with other more complex MoE, and outperforms non-mixture baselines. This showcases the merits of even a simple single-gate MoE, and motivates further exploration in this area.

1 Introduction

Neural networks are designed to extract a fixed set of exhaustive features for any given image. However, images exhibit varying levels of complexity, from simple cases such as single objects on a white background to images with clutter and difficult camera angles. Treating both of these cases equally can be wasteful from an efficiency perspective: This intuition has given rise to very active research in the fields of conditional computing [3] and early-exiting [19, 44, 48]. In conditional computing, subparts of the network are turned on or off dynamically based on the input image. This allows to increase the network's capacity at training time without affecting the computational cost at inference. The same conditional behavior applies in early exiting but across the depth dimension: The prediction can be finalized early in the network for simple images, avoiding unnecessary further computing.

In particular, Mixture of Experts (MoE) have gained a lot of traction in recent years for conditional computing. For instance, transformers with a massive number of parameters

are now becoming the new normal for natural language processing [7, 11, 25, 39]. Similar models are also starting to emerge in computer vision, leveraging extremely large datasets and numerous routing decisions [39]. The success of these large-scale conditional models begs the question whether similar results are also achievable for datasets and architectures of a smaller scale, more commonly used by practitioners (e.g. ResNet-18 on ImageNet). In this work, we introduce three ingredients to make simple single-gate MoE competitive with other state-of-the-art MoE, across a variety of architectures and dataset sizes. In particular, our training pipeline remains efficient and stable in all cases, in contrast to more complex dynamic routings [51], and avoids introducing new ad-hoc losses [52]. Specifically, we make the following contributions:

- In Section 2.2, we introduce a single-gate MoE that consistently outperforms its non-mixture counterparts on various architectures. A key improvement in our proposed model is a *base network* branch whose features facilitate the initial expert selection. We also show that this base model acts as an excellent regularizer when ensembled with specialized experts, improving the overall performance.
- In Algorithm 1, we also formulate a *simple and efficient* training procedure for the model, which is both stable and asynchronous. These results indicate that simple single-gate MoE is a promising direction to enable conditional computing for both small training- and inference- computational budgets.
- Finally, in Section 2.3, we propose and evaluate a simple threshold rule to dynamically adapt the computational budget at inference, without retraining, which combines early-exiting through the base model and selecting a dynamic number of experts per sample.

2 Proposed Model

2.1 The Mixture of Experts Setup for Image Classification

A Mixture of Experts (MoE) consists of a set of K *experts*, $(e_k)_{1\dots K}$, each outputting a distribution over the target classes; The execution of these experts is conditioned by the *gate* g (or *router*), which outputs a probability distribution over the set of experts. The total likelihood of the model on the training dataset $\mathcal{D}$, which we want to maximize, is expressed as:

$$\mathcal{L}(D) = \mathop{\mathbb{E}}_{(x,y)\sim D}\left[\sum_{k=1}^{K} g(k|x)\, e_k(y|x)\right] \tag{1}$$

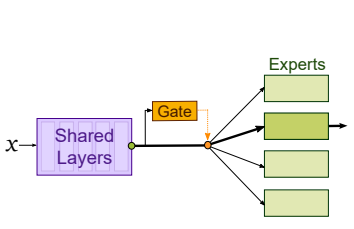
A successful MoE relies on the gate learning a decomposition of the input space across K clusters, such that experts specialize on the resulting subsets; The key underlying assumption is that this compositional approach outperforms a single model trained on the entire dataset. At inference time, the gate is thresholded, such that only one - or few - experts are executed, to control the accuracy/efficiency trade-off. Unfortunately, the standard MoE suffers from three major issues: **(i)** Because the experts only have a local view of the training set, regulated by the gate, their mixture is more prone to overfitting than a single model trained on the whole dataset [40]. **(ii)** Jointly training the gate and experts raises a chicken-and-egg problem: The gate has to route samples to the experts most likely to classify them successfully, but weaker experts need data to improve. This problem often leads to the gate collapsing, i.e., only feeding input samples to very few experts, which defeats the purpose of using an MoE in

the first place. **(iii)** The initial data subsets defined by the gate strongly influences the expert training. Thus, a naive random initialization may even further worsen the gate collapse issue if, for instance, an expert is heavily favored over others at initialization.

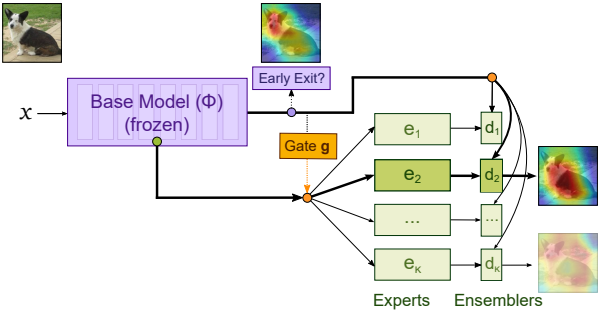
In this work, we propose two key changes to the MoE framework to alleviate the aforementioned issues and improve performance of simple single-gate MoE models. **First**, we introduce a novel generic knowledge branch, which we refer to as the *base model*: This module is trained on the whole dataset, and we use it **(i)** to tackle potential overfitting: It acts as a form of regularization for the experts by ensembling their outputs with this initial base prediction; **(ii)** to initialize the gate, using the feature space induced by the base model for clustering the training samples. and **(iii)** as an early-exiting branch that avoids executing any expert when not necessary, and is conditionally activated based on the input image. **Second**, we describe a simple, lightweight training scheme that first initializes the experts' subsets by clustering the base model's embeddings and then keeps the gate and experts independent during training in order to avoid the gate collapse issue.

In Section 2.2, we describe the proposed model's key components and training scheme; The model architecture is summarized in Figure 1. Then, in Section 2.3 we describe a simple conditional computing mechanism to obtain even more computationally-efficient models.

2.2 Model Summary



(a) Single-gate MoE, e.g. [14, 34, 50, 52]



(b) Proposed model

Figure 1: (*left*) MoE define a gate, g, that selects (●) which expert to execute based on the current representation (●) of input x. At inference, a unique expert is picked (in **bold**). (*right*) our proposed architecture maintains a full-depth base model, ϕ, which is (i) ensembled with the expert output, (ii) used as inputs to the experts and gate, and (iii) acts as an early exit (●) at inference. Grad-CAM [12] visuals reveal that the selected expert focuses on fine-grained details, while the base model attends to general features. The other non-selected experts produce poorly focused activation maps.

Architecture. The **base model** ϕ is a simple, ideally lightweight, network trained on the whole dataset, and is executed for every input. Its purpose is multi-fold: First, it is ensembled with the selected expert. While previous works [34, 50] often ensemble specialized experts together in MoE, we show in our experiments that ensembling one expert with this non-specialist branch is consistently more beneficial. Second, the base model acts as an early exit output at inference time, avoiding redundant expert computations for the easier samples (see Section 2.3). Finally, we reuse the early layers of the base model as inputs to the gate and the experts which allows us to reduce computational load even further.

The **gate** g is a simple linear layer taking as input the pre-logits of the base model. At training time, it outputs a probability distribution over experts, $g(k|x)$, allowing for direct

backpropagation through these weights. At inference, we only select and execute the most probable expert, i.e., $g_{\text{test}}(k|x) = \mathbb{1}(k = \arg\max_{k'} g(k'|x))$; $\mathbb{1}(\cdot)$ being the indicator function. We also discuss in Section 2.3 how we can dynamically select the number of active experts, rather than always defaulting to the top-1 expert.

Experts are neural networks whose input is an intermediate feature map of the base model. This design choice yields two benefits: **(i)** The experts' early features are shared and frozen, which reduces the number of trainable parameters and reduces the risk of experts overfitting (in particular when training on small datasets); and **(ii)** this allows the model to reuse computations from the base model at inference time, further improving efficiency.

Finally, **ensemblers** are shallow neural networks, one for each expert, combining outputs of the base model and the expert selected by the gate. We experiment with both stacking and bagging ensembling methods. In the text, we also refer to $e'_k(y|x) = d_k(\phi(y|x); e_k(y|x))$ as the classification output of the ensembler d_k, which ensembles the k-th expert and base model ϕ.

Training Procedure. We summarize our asynchronous training scheme in Algorithm 1, in which the gate and experts are trained independently in parallel. This training procedure relies on three key insights: **First**, to avoid gate collapse, we keep the gate weights fixed while training the experts. This makes the model heavily dependent on the gate initialization. Thus, to define a meaningful initial gate g_0, we cluster the pretrained base model's embeddings using K-means [29]. A similar initialization scheme has been used in the hierarchical classification literature [14, 34]; In contrast, we do not restrict this initial clustering step to be a hard assignment to a unique expert, nor to be on a per-class basis.

A **second** issue stems from uncalibrated outputs [15, 33]: Training an ensembler d_k jointly with its expert e_k often leads to d_k heavily favoring the base model, preventing the expert from specializing. This behavior is likely due to the base model being overly confident on many training samples: In fact, this is particularly apparent on small datasets where the base model is already close to perfectly fitting the training set, e.g., on CIFAR-100. To avoid this problem, we only start training d_k *after* fully training the corresponding expert e_k.

Thirdly, because the experts are initialized with the base model's pretrained weights, but are then trained on a specialized subset of the data given by the gate, they might "forget" classes they never see. This is similar to *catastrophic forgetting* [22, 37]. While the proposed ensemblers partially alleviate this issue by providing additional regularization, we find that it is often beneficial to also route non-assigned samples the experts: Specifically, in step 4 and 5 of Algorithm 1, the gate g_0 is "smoothed" using the transformation: $\Gamma : \cdot \mapsto \text{clip}(\cdot, \gamma, 1.)$, where γ is a hyperparameter. We experimented with **(i)** using the smoothed gate weights to re-weight the loss of all samples, including negative ones (as portrayed in **Algorithm 1**) or **(ii)** using these gate weights as sampling probabilities when forming the training batch. Both yield similar results, and while (i) is simpler to implement, we find that (ii) is more practical for large datasets, as it often leads to faster convergence, hence reduced training times.

An alternative joint training scheme. We also consider extending the training scheme to handle joint end-to-end training of the gate and experts by using the Expectation Maximization (**EM**) algorithm [8] to alleviate the "chicken-and-egg" problem. EM alternates between two steps: **(E)** computing new gate weights, updated based on the current experts' performance, and **(M)** separately training the experts according to this new assignment, while forcing the gate to match it. Taking into account training costs, it is beneficial to keep a low number of **E** steps (N_E) as every update of the posterior requires synchronization across all experts. In fact, one can show that Algorithm 1 is equivalent to setting $N_E = 0$. In prac-

Algorithm 1 Proposed asynchronous training scheme

Require: Training dataset $\mathcal{D}$, base model ϕ, gate g, experts $e_{1...K}$, ensemblers $d_{1...K}$

1. Train the **base model** ϕ on dataset $\mathcal{D}$ (or use an off-the-shelf pretrained model)
2. Cluster the base model embeddings using K-means, obtaining centroids $c_{1...K}$

Define the **initial gate** $g_0 : (x,k) \mapsto m(\phi(x), c_k)$ where m is a similarity function and is such that the weights across clusters sum to 1 for a given sample x.

3. Train the **gate** g by minimizing the KL divergence $\mathrm{KL}(g_0, g)$

for $k = 1$ to K (in parallel) **do**

Initialize the k-th expert from the base model's weights: $\theta_{e_k} \leftarrow \theta_\phi$

4. Train **expert** e_k following (1), i.e., by maximizing $\sum_{(x,y)\sim D} g_0(k|x)\ e_k(y|x)$
5. Train **ensembler** d_k by maximizing $\sum_{(x,y)\sim D} g_0(k|x)\ d_k(\phi(y|x); e_k(y|x))$

tice, we observe that larger values N_E can lead to higher accuracies (e.g., on tiny-ImageNet: **+1.37%** without ensemblers, and **+0.43%** with ensemblers). However, the improved performance is often not worth the higher training costs, hence we only report results using **Algorithm 1** in our experiments section. We describe the derivation of the EM variant and experiments in more details in the **supplemental material**.

2.3 Anytime Inference via Early-Exiting and Dynamic Ensembling

By design, our framework integrates a straightforward option for early-exiting by directly outputting the base model's predictions in easy cases, to improve computational efficiency further. Following previous early-exiting literature [19, 44, 49], our model decides whether to early-exit or to execute the gate-selected expert by thresholding the base model's confidence at inference time. We also consider other early-exit designs in the **supplemental material**. Additionally, it is clear from Equation 1 that MoE can be viewed as an ensemble of experts weighted by the gate, rather than using only the top-1 expert as is usually done for efficiency purposes. Similar to [50], we propose to threshold the gate outputs to determine which experts to include dynamically at inference.

In order to combine both the early-exiting and expert ensembling behaviors under a unique thresholding rule, we introduce the quantity $\alpha_{\mathbf{k}}(\mathbf{x}) = \mathbf{g}(\mathbf{k}|\mathbf{x})(\mathbf{1} - \max_{\mathbf{y}} \phi(\mathbf{y}|\mathbf{x}))$. From a probabilistic perspective, $\alpha_k(x)$ can be interpreted as the joint probability that the sample x is not early-exited, *and* that the gate routes x to the k-th expert. Intuitively, if this quantity is below a certain threshold for all experts, it means that the base model has a high confidence and the gate does not confidently route the sample to any expert; thus we should early exit. Thus, given a trained gate g and experts (with their ensemblers) e'_k, we define the **anytime model** $p^{\text{at-}\tau}$, which combines both early-exiting and dynamic experts ensembling, as follows:

$$ee(x) = 1 \text{ iff } \forall k \in [1,K],\ \alpha_k(x) < \tau \tag{2}$$

$$p^{\text{at-}\tau}(y|x) = ee(x)\phi(y|x) + (1 - ee(x)) \sum_{k=1}^{K} \mathbb{1}(\alpha_k(x) \geq \tau)\ g(k|x)\ e'_k(y|x) \tag{3}$$

where $\mathbb{1}(\cdot)$ is the indicator function, and $\tau \in [0,1]$ is a hyperparameter. We show in experiments that varying τ allows the model to quickly achieve a wide range of computational budgets at inference time, without any retraining.

3 Related Work

Conditional computing aims to learn a sparse connectivity pattern conditioned on the input sample. To achieve such pattern, many works add a mixture of expert layers at several stages throughout the network [5, 10, 31, 38, 46, 47, 51]. Unlike single-gate MoE, the increased number of routing decisions incurs some practical drawbacks: (i) At inference, a new submodel has to be loaded in memory for every routing decision, which becomes increasingly cumbersome as the number of gates increases and (ii) all gates and experts have to be trained synchronously which leads to huge models to train and training instabilities. This often leads to complex training pipelines, e.g. relying on reinforcement learning [5, 38, 46] to learn the routing mechanism. More recently, [51] has proposed a simpler k-means-like routing mechanism that evolves during training via moving average. However, they also report that the training pipeline requires large batch sizes, and is prone to mode collapse.

Simpler **single-gate mixture of experts** have also been successfully applied to neural networks for various applications such as image classification [1], detection [24] , retrieval [14] and scene recognition [20]. More recently, [52] has shown that such models can achieve significant accuracy/efficiency gains in the large-scale regime. However, their training pipeline relies on several unclear heuristics. Orthogonal to these, hierarchical classification is a subclass of MoE, in which the routing is learned on a *per-class* basis, aiming to route all samples of a ground-truth class to the same expert. Several works [4, 13, 30, 35] directly leverage an external class taxonomy (such as WordNet [32]). A follow-up line of thought extracts such information from a pretrained classifier [34, 50], or even learns the optimal taxonomy jointly with the image representations [2, 28]. Such models have been shown to improve the efficiency/accuracy trade-off in classification tasks. However, this class-based routing is a limiting assumption, and per-sample routing has been shown to outperform hierarchical classification models when correctly parametrized [1, 20].

Finally, MoE can be seen as an **ensembling** technique whose weights are learned by the gate. While it is common to assume each sample is routed to a unique expert to maximize efficiency, some works [14, 34, 50] have considered combining several experts to boost accuracy. In contrast, we show that combining one specialized expert with the generic knowledge base model with simple ensemble methods such as averaging or linear stacking [43] is generally more efficient than ensembling multiple specialized experts.

4 Experiments

We perform experiments on datasets of different scales: CIFAR-100 [23], tiny-ImageNet [26] (a downscaled subset of ImageNet with 200 classes and 110k images), and ILSVRC2012 [41].

We use ResNets [16] with different depths as our main backbone architecture. For CIFAR-100 and tiny-ImageNet, we use a modified variant of ResNets which eliminates the first two downscaling operations (strided convolution and max-pooling), commonly used in the literature [21, 45, 51]; We dub it "*tiny-ResNet*" or **tr** for short. We follow previously established training pipelines to train our baseline models, specifically [9] for CIFAR-100 and [27] for tiny-ImageNet.

For ImageNet, we additionally perform experiments on MobileNetv3-small [17], and use the standard checkpoints provided in torchvision's model zoo [36] as base models. In all of our experiments, we initialize the experts with pretrained weights from the base model and train them with the same hyperparameters and data augmentations as the baseline, although

Base Model	Expert	top-1 acc	MACs (x 1e9)	#params x 1e7 inference	#params x 1e7 trainable
tr18 baseline	-	77.95	0.56	1.12	1.12
tr10	tr10	77.96 ± 0.20	0.37	0.96	9.29
	tr18	78.78 ± 0.22	0.52	1.55	21.1
tr34 baseline	-	78.60	1.16	2.13	2.13
tr18	tr10	79.78 ± 0.05	0.67	1.58	9.29
	tr18	79.90 ± 0.22	0.82	2.17	21.1
tr50 baseline	-	80.10	1.30	2.37	2.37
tr34	tr10	80.48 ± 0.17	1.28	2.59	9.29

Base Model	Expert	top-1 acc	MACs (x 1e9)	#params x 1e7 inference	#params x 1e7 trainable
tr18 baseline	-	60.42	2.22	1.13	1.13
tr10	tr10	60.58 ± 0.04	1.48	0.96	9.39
	tr18	63.38 ± 0.05	2.09	1.55	21.2
tr34 baseline	-	63.39	4.64	2.14	2.14
tr18	tr10	64.46 ± 0.05	2.69	1.59	9.39
	tr18	66.26 ± 0.05	3.30	2.18	21.2
tr50 baseline	-	63.24	5.19	2.39	2.39
tr34	tr10	66.42 ± 0.15	5.11	2.60	9.39

Table 1: Main CIFAR-100 (*left*) and tiny-ImageNet (*right*) results. Each number is reported over three random seeds. All settings have 20 experts, whose first three blocks are the (frozen) layers of the base model. We report accuracy and efficiency metrics (number of operations and number of parameters) across various base and experts architectures.

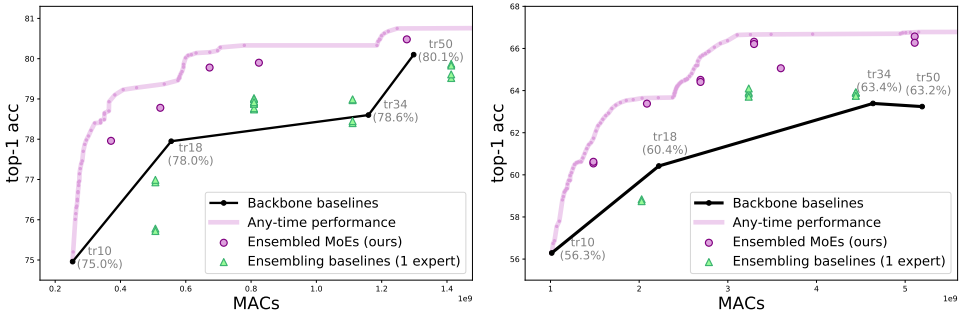


Figure 2: Accuracy vs MACs performance of our "anytime" variant models using a simple thresholding rule on CIFAR-100 (*left*) and tiny-ImageNet (*right*).

using fewer training iterations (200 epochs for CIFAR100, 100 for tiny-ImageNet and 40 for ImageNet). The features of the base model are kept frozen.

In this section, to assess the benefits of our proposed model and training scheme, we compare our proposed method to **(i)** the backbone models at different depths, **(ii)** an ensembling baseline with equivalent computational cost, **(iii)** hierarchical classification, and **(iv)** two recent dynamic routing works [47, 51]. We also report results of an ablation experiment on using different ensembling methods. Finally, we report detailed hyperparameters used for the experiments, and further ablations in the supplemental material.

4.1 Results on Small and Medium-scale Datasets

We first report results on CIFAR-100 and tiny-ImageNet for tiny-ResNets of different depths in Table 1. All results are reported with 20 experts, branching off the base model after the third residual block, and *without any early-exiting or dynamic ensembling*. These results show that even simple single-gate MoE can significantly improve the efficiency/accuracy trade-off over standard CNNs: Our proposed method consistently outperforms the backbone network for an equivalent MAC count. The only downside is that MoE generally has a higher parameter count at training time. Nevertheless, our asynchronous training scheme allows us to train experts independently across multiple devices efficiently. We also observe that using a deeper base model is often more beneficial than using deeper experts in terms of accuracy.

Comparison to ensembling. A natural baseline to compare to is to ensemble the base model with a unique "expert" trained on the whole dataset: The resulting model has the same cost as its MoE counterpart, minus the negligible cost of the linear layer gate. We also analyze the impact of the number of experts on the model: Adding more experts leads to higher specialization, but also more potential routing errors; thus, it is not evident that increasing the number of experts benefits the model. We report the corresponding results in Table 2: All MoE outperform the ensembling baseline, which shows the benefit of specialized experts. Hower, the impact of going from 10 to 20 experts is minimal: The benefit of splitting the data into specialized subsets starts to fade above 10 experts for these datasets.

tr18-tr18	CIFAR100	tiny-ImageNet
base model	77.95	60.42
baseline (1 expert)	78.99 ± 0.29	63.83 ± 0.08
5 experts	79.67 ± 0.14	65.42 ± 0.15
10 experts	79.84 ± 0.13	65.72 ± 0.17
20 experts	**79.90** ± 0.22	**66.26** ± 0.05

Table 2: Impact of the number of experts on the model accuracy and comparison to the ensembling baseline (equivalent to using only one expert in our model).

Anytime Inference Models. We explore the effect of the anytime inference model introduced in Section 2.3. To fully understand the scope of this dynamic behavior, we evaluate the accuracy of all models across various thresholds. We then plot the convex envelope of this set of curves, as shown in Figure 2. We observe that dynamically deciding for each sample whether to early exit through the base model or to use one or more experts consistently improves the overall accuracy/efficiency trade-off. Furthermore, this simple thresholding rule allows us to quickly adapt the model's computational budget without retraining.

4.2 Results on ImageNet

Main results. We report results on ImageNet experiments for ResNet and MobileNet backbones in Table 3. Our previous observations still hold: The MoE model outperforms both the ensembling baseline and the backbone, and early-exiting based on the base model's confidences helps further reduce computations for a limited drop in accuracy.

We also note that, unlike ResNet18, ResNet34's MobileNetv3's performance start to saturate and even slightly decreases with more experts. This behavior implies that the optimal number of experts is not only a property of the dataset but also of the architecture of both the experts and the base model.

ResNet18	None	$\tau = 0.75$	$\tau = 0.5$
baseline (1 expert)	71.50	71.50	71.13
4 experts	72.17	72.11	71.68
20 experts	**72.38**	**72.38**	**71.73**
MACs	2.64e9	2.18e9	2.03e9

(a) ResNet18 base model (**69.76**% accuracy, 1.82 GMACs) and experts

None	$\tau = 0.75$	$\tau = 0.5$
74.33	74.29	74.00
75.03	74.89	74.47
75.05	**74.92**	**74.56**
5.64e9	4.42e9	4.06e9

(b) ResNet34 base model (**73.31**%, 3.66 GMACs) and experts

None	$\tau = 0.75$	$\tau = 0.5$
68.06	68.13	68.15
68.60	**68.59**	68.44
68.58	68.53	**68.46**
8.13e7	6.83e7	6.36e7

(b) MobileNetv3-small base (**67.67**%, 5.65e7 MACs) and experts

Table 3: Main results on ImageNet. Experts share 3 layers in the ResNet experiments, and 8 in the MobileNet ones (half of the model in both case). We report our main results in the first column, without any early-exiting or dynamic ensembling. In the following columns, we report additional results with early-exiting obtained with different values of the threshold τ on the base model confidence.

Comparison to Dynamic Routing baselines. In this section, we compare our model to two recent dynamic routing works. In Table 4 (*left*) we compare to DeepMoE [47] on Imagenet: The model is a two-times wider ResNet-18 trained end-to-end with a sparsity constraint forcing the router in each layer to only activate roughly half of the channels for each sample. In Table 4 (*right*), we compare to RMN [51] on tiny-ImageNet: Each residual block is replicated 8 times (a total of 8^4 different computational paths), and for each, a routing based on a moving average of initial k-means centroids is learned. We could not compare directly to RMN on ImageNet as [51] uses a modified ResNet architecture including additional Squeeze&Excite [18] layers. Nevertheless, in terms of relative accuracy improvement and MACs, we observe that our model yields comparable trade-offs, despite using a single gate, hence significantly fewer computational paths. Furthermore, in contrast to both approaches, we can easily reach various computational budgets without retraining via early-exiting.

ImageNet Resnet-18	gates	acc.	MACs x1e9	#params (train) x1e9
ours	1	72.17	2.64	5.10
+early-exit $\tau = 0.75$	1	72.11	2.18	5.10
DeepMoE [47]	17	70.95	1.81	7.02

tiny-ImageNet `tr18`	gates	base model	acc.	MACs	#params (train)
ours (10 exp)	1	60.42	64.66	2.69e9	5.98e7
+ early-exit ($\tau = 0.75$)	1	60.42	64.58	2.44e9	5.98e7
RMN (no SE) [51]	5	61.78	64.30	2.22e9	9.02e7

Table 4: Comparison to recent dynamic routing literature. On the left, we compare our 4-experts ImageNet model with Wide-DeepMoE-18 from [47]. On the right, we compare our `tr18-tr10` 10 experts model with the tiny-ResNet18-based model of [51], excluding additional Squeeze&Excite layers. For [51], we also report the corresponding base model accuracy as we could not reproduce their baseline training results to use in our experiments.

4.3 Ablation experiments

In the supplemental material, we report further ablation experiments on (i) different early-exiting procedures an (ii) an alternative joint training based on Expectation-Maximization.

Per-sample vs per-class clustering initialization. Hierarchical classifiers are single-gated models whose routing is learned on a strict *per-class* basis. In the literature, the class taxonomy is either based on external knowledge [4, 13, 30, 35], clustering pretrained embeddings [34, 50], or jointly learned alongside the features [2]. We implement a per-class variant of our model, following the clustering process of [34, 50].

top-1 acc	w/o ensembling	w/ ensembling
per-sample (**ours**)	**63.11** ± 0.11	**65.72** ± 0.10
per-class	62.48 ± 0.13	63.85 ± 0.08
per-class + *oracle*	68.8 ± 0.10	67.99 ± 0.04

Table 5: Comparison to hierarchical classification. We introduce a per-class variant of our model following [34, 50], and an oracle variant in which the gate follows the true class-to-expert distribution. Results are reported on tiny-ImageNet with 10 experts, in the `tr18-tr18` configuration.

In Table 5, we show that per-sample routing always outperforms its per-class counterpart. Furthermore, we observe a significant accuracy gain if we re-evaluate the per-class model using the samples' ground-truth classes as an oracle for predicting the correct expert. This indicates that learning the per-class routing is the main bottleneck. In fact, a sample (x, y) incorrectly mapped to an expert that has never seen class y is very detrimental to accuracy, even if this is partially compensated by the effect of ensembling, In contrast, per-sample routing allows several experts to gain knowledge about the same class and introduces a more flexible notion of diversity among experts.

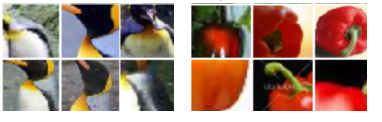
Analysing the Ensemblers' performance In Table 6, we compare results for different ensembling scheme with equivalent computation costs: Even without ensembling, MoE outperforms the base model. Nevertheless, all ensembling methods perform strictly better than trusting the selected expert only. Second, we observe that **stacking** and **bagging** always outperform ensembling the top-2 experts, which shows the benefit of ensembling with the base model rather than another specialized expert. We use **bagging** in our experiments as it is simpler, non-parametric, and does not require training.

base - expert	tr10-tr10	tr10-tr18	tr18-tr18
base model	56.29	56.29	60.42
no ensembling	57.80	59.53	63.82
top-2 experts	58.69	60.55	64.19
stacking	**60.84**	64.63	66.29
bagging	60.54	**64.77**	**66.32**

Table 6: Impact of the ensembler design on accuracy (tiny-ImageNet, 20 experts)

Qualitative Analysis of The Experts' Specialization Pattern. Finally, we qualitatively analyze the experts' behavior: For each sample, we record which expert reaches minimal cross-entropy loss on that given sample. We then display the resulting class distribution across experts of the whole training set (see Appendix 3): The experts do end up specializing to specific subsets of the data, although, in contrast to hierarchical classification, many classes are still clearly split across several experts. Furthermore, most of these specialization patterns are consistent across the number of experts. Finally, while some experts clearly account for more classes than others, no expert is ever fully inactive. Looking more closely at the routed samples, we also see that the router uncovers natural intra-class variations, as illustrated for the class `king penguin` in Figure 3: The class `king penguin` is split across (i) close-up images where the orange beak is very visible, and end up being mapped to the same expert as oranges, bellpepper etc, which it is often confused with, and (ii) far away images which are instead grouped with other animal classes.

(a) The class `king-penguin` (left) co-occurs with other animals (right) for full-view images...



(b) ...but is often grouped with e.g., `bell pepper` when the image is a close-up view of its orange beak.

Figure 3: Per-sample routing uncovers meaningful intra-class modes.

5 Conclusions

In this work, we revisit the single-gate Mixture of Experts (MoE) for convolutional architectures. Specifically, we augment MoE with a novel ensembling scheme and a simple asynchronous and stable training pipeline leveraging a clustering-based initialization. Our model consistently reaches higher accuracy than hierarchical classifiers and a 1-expert ensembling baseline, revealing the benefits of training specialized experts with per-sample routing. Moreover, maintaining the base model as an independent branch allows us to further save computations at inference time using a simple threshold-based conditional rule.

Finally, our model is competitive with recent multi-layer MoE dynamic routing works, despite a smaller number of routers and experts, and provides a more lightweight and stable training pipeline. In the future, we plan to further improve our model's training efficiency by investigating different sampling strategies based on the gate outputs.

References

[1] Karim Ahmed and Lorenzo Torresani. BranchConnect: Image categorization with learned branch connections. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018.

[2] Karim Ahmed, Mohammad Haris Baig, and Lorenzo Torresani. Network of experts for large-scale image categorization. In *European Conference on Computer Vision (ECCV)*, 2016.

[3] Leonard Nicholas Bengio, Yoshua and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *ArXiv*, abs/1308.3432, 2013.

[4] Luca Bertinetto, Romain Mueller, Konstantinos Tertikas, Sina Samangooei, and Nicholas A. Lord. Making better mistakes: Leveraging class hierarchies with deep networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[5] Shaofeng Cai, Yao Shu, and Wei Wang. Dynamic routing networks. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[6] Xinshi Chen, Hanjun Dai, Yu Li, Xin Gao, and Le Song. Learning to stop while learning to predict. In *International Conference on Machine Learing (ICML)*, 2020.

[7] Aidan Clark, Diego de Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake A. Hechtman, Trevor Cai, Sebastian Borgeaud, George van den Driessche, Eliza Rutherford, Tom Hennigan, Matthew Johnson, Katie Millican, Albin Cassirer, Chris Jones, Elena Buchatskaya, David Budden, Laurent Sifre, Simon Osindero, Oriol Vinyals, Jack W. Rae, Erich Elsen, Koray Kavukcuoglu, and Karen Simonyan. Unified scaling laws for routed language models. *ArXiv*, abs/2202.01169, 2022.

[8] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the EM. 1977.

[9] Terrance Devries and Graham W. Taylor. Improved regularization of convolutional neural networks with cutout. *ArXiv*, abs/1708.04552, 2017.

[10] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. Learning factored representations in a deep mixture of experts. In *Workshops of the International Conference on Learning Representations (ICLR)*, 2014.

[11] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *ArXiv*, abs/2101.03961, 2021.

[12] Gildenblat, J. and contributors. Pytorch library for cam methods. https://github.com/jacobgil/pytorch-grad-cam, 2021.

[13] Wonjoon Goo, Juyong Kim, Gunhee Kim, and Sung Ju Hwang. Taxonomy-regularized semantic deep convolutional neural networks. In *European Conference on Computer Vision (ECCV)*, 2016.

[14] Sam Gross, Marc'Aurelio Ranzato, and Arthur D. Szlam. Hard mixtures of experts for large scale weakly supervised vision. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[15] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learing (ICML)*, 2017.

[16] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[17] Andrew G. Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for MobileNetV3. In *International Conference on Computer Vision (ICCV)*, 2019.

[18] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-Excitation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 2020.

[19] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Q. Weinberger. Multi-scale dense networks for resource efficient image classification. In *International Conference on Learning Representations (ICLR)*, 2018.

[20] Hyo Jin Kim and Jan-Michael Frahm. Hierarchy of alternating specialists for scene recognition. In *European Conference on Computer Vision (ECCV)*, 2018.

[21] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. Puzzle Mix: Exploiting saliency and local statistics for optimal mixup. In *International Conference on Machine Learing (ICML)*, 2020.

[22] James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. In *Proceedings of the National Academy of Sciences*, 2017.

[23] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[24] H. Lee, S. Eum, and H. Kwon. ME R-CNN: Multi-expert R-CNN for object detection. *IEEE Transactions on Image Processing*, 29:1030–1044, 2020.

[25] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. GShard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations (ICLR)*, 2021.

[26] Fei Li, Andrej Karpathy, and Justin Johnson. Tiny-ImageNet visual recognition challenge. https://www.kaggle.com/c/tiny-imagenet, 2017.

[27] Siyuan Li, Zicheng Liu, Di Wu, Zihan Liu, and Stan Z. Li. Boosting discriminative visual representation learning with scenario-agnostic mixup. *ArXiv*, abs/2111.15454, 2021.

[28] Xiaoni Li, Yucan Zhou, Yu Zhou, and Weiping Wang. MMF: Multi-task multi-structure fusion for hierarchical image classification. In *International Conference on Artificial Neural Nertworks (ICANN)*, 2021.

[29] J. MacQueen. Some methods for classification and analysis of multivariate observations. 1967.

[30] Marcin Marszalek and Cordelia Schmid. Constructing category hierarchies for visual recognition. In *European Conference on Computer Vision (ECCV)*, 2008.

[31] Mason McGill and Pietro Perona. Deciding how to decide: Dynamic routing in artificial neural networks. In *International Conference on Machine Learing (ICML)*, 2017.

[32] George A. Miller, Richard Beckwith, Christiane D. Fellbaum, Derek Gross, and Katherine J. Miller. Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography*, 1990.

[33] Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Ann Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the calibration of modern neural networks. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[34] Ravi Teja Mullapudi, William R. Mark, Noam M. Shazeer, and Kayvon Fatahalian. HydraNets: Specialized dynamic architectures for efficient inference. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[35] Joan Puigcerver, Carlos Riquelme, Basil Mustafa, Cédric Renggli, André Susano Pinto, Sylvain Gelly, Daniel Keysers, and Neil Houlsby. Scalable transfer learning with expert models. In *International Conference on Learning Representations (ICLR)*, 2021.

[36] Pytorch. Torchvision model zoo. https://pytorch.org/vision/stable/models.html.

[37] Vinay V. Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. In *International Conference on Learning Representations (ICLR)*, 2021.

[38] Yongming Rao, Jiwen Lu, Ji Lin, and Jie Zhou. Runtime network routing for efficient image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*, 2019.

[39] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[40] Clemens Rosenbaum, Ignacio Cases, Matthew Riemer, and Tim Klinger. Routing networks and the challenges of modular and compositional computation. *ArXiv*, abs/1904.12774, 2019.

[41] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[42] Simone Scardapane, Danilo Comminiello, Michele Scarpiniti, Enzo Baccarelli, and Aurelio Uncini. Differentiable branching in deep networks for fast inference. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2020.

[43] Joseph Sill, Gábor Takács, Lester W. Mackey, and David Lin. Feature-weighted linear stacking. *ArXiv*, abs/0911.0460, 2009.

[44] S. Teerapittayanon, B. McDanel, and H. T. Kung. BranchyNet: Fast inference via early exiting from deep neural networks. In *International Conference on Pattern Recognition (ICPR)*, 2016.

[45] A. F. M. Shahab Uddin, Mst Sirazam Monira, Wheemyung Shin, TaeChoong Chung, and Sung-Ho Bae. SaliencyMix: A saliency guided data augmentation strategy for better regularization. In *International Conference on Learning Representations (ICLR)*, volume abs/2006.01791, 2021.

[46] Xin Wang, Fisher Yu, Zi-Yi Dou, and Joseph Gonzalez. SkipNet: Learning dynamic routing in convolutional networks. In *European Conference on Computer Vision (ECCV)*, 2018.

[47] Xin Wang, Fisher Yu, Lisa Dunlap, Yian Ma, Yi-An Ma, Ruth Wang, Azalia Mirhoseini, Trevor Darrell, and Joseph Gonzalez. Deep mixture of experts via shallow embedding. In *Uncertainty in Artificial Intelligence (UAI)*, 2019.

[48] Zuxuan Wu, Tushar Nagarajan, Abhishek Kumar, Steven J. Rennie, Larry S. Davis, Kristen Grauman, and Rogério Schmidt Feris. BlockDrop: Dynamic inference paths in residual networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[49] Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy J. Lin. DeeBERT: dynamic early exiting for accelerating BERT inference. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020.

[50] 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 *International Conference on Computer Vision (ICCV)*, 2015.

[51] Kaipeng Zhang, Zhenqiang Li, Zhifeng Li, Wei Liu, and Yoichi Sato. Neural routing by memory. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[52] Yikang Zhang, Zhuo Chen, and Zhao Zhong. Collaboration of experts: Achieving 80% top-1 accuracy on ImageNet with 100m FLOPs. *ArXiv*, abs/2107.03815, 2021.