

Hierarchy of Alternating Specialists for Scene Recognition

Hyo Jin Kim and Jan-Michael Frahm

Department of Computer Science, University of North Carolina at Chapel Hill
{hyojin, jmf}@cs.unc.edu

Abstract. We introduce a method for improving convolutional neural networks (CNNs) for scene classification. We present a hierarchy of specialist networks, which disentangles the intra-class variation and inter-class similarity in a coarse to fine manner. Our key insight is that each subset within a class is often associated with different types of inter-class similarity. This suggests that existing network of experts approaches that organize classes into coarse categories are suboptimal. In contrast, we group images based on high-level appearance features rather than their class membership and dedicate a specialist model per group. In addition, we propose an alternating architecture with a global ordered- and a global orderless-representation to account for both the coarse layout of the scene and the transient objects. We demonstrate that it leads to better performance than using a single type of representation as well as the fused features. We also introduce a mini-batch soft k -means that allows end-to-end fine-tuning, as well as a novel routing function for assigning images to specialists. Experimental results show that the proposed approach achieves a significant improvement over baselines including the existing tree-structured CNNs with class-based grouping.

Keywords: deep learning · hierarchy of specialists · scene recognition

1 Introduction

Accurately identifying the background in an image (e.g. beach, mountains, candy store) is an important task in computer vision because it provides us with strong contextual information as to what is happening in the scene. The major challenge that needs to be addressed is the severe intra-class variation and inter-class similarity. Not only there are many visually diverse instances within one scene category (e.g. Notre-Dame de Paris vs. Saint Basil's Cathedral), but there is also a significant visual overlap between different scene categories (e.g. airports vs. modern train stations). Several approaches have been proposed to address this problem by designing or learning better visual features [8,9,18,45,50,59]. Newer end-to-end deep neural networks were able to achieve state-of-the-art classification accuracy [1,63]. However, it becomes increasingly hard to find a distinctive representation when the classes become visually nearly indistinguishable as the number of classes increases [40]. Downweighting the representations

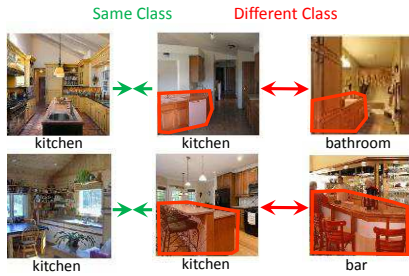


Fig. 1. Examples of intra-class variation and inter-class similarity. While base cabinets and bars characterize the kitchen class, they cause overlap with other classes with similar furnishings.

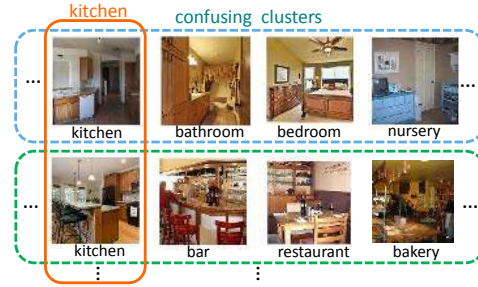


Fig. 2. There are subsets of images in each class that are often confused with those of other classes. We discover *confusing clusters* in the feature space to disentangle intra-class variation and inter-class similarity.

for commonly shared visual elements can help reduce the inter-class similarity. However, these elements are sometimes the key to distinguish a class from others, as illustrated in Fig. 1.

Thus, a sensible way to handle this issue is to apply a divide and conquer [48] strategy to dedicate different CNNs to separable subproblems. Existing methods organize classes into coarse categories, either based on the semantic hierarchy [12,15,22,58] or the confusion matrix of a trained classifier [53,57]. However, we observe that there are multiple modes of intra-class appearance variation, and that each of these modes typically causes overlap with different subsets of categories. As depicted in Fig. 2, some images of a kitchen with cabinets can be confused with a bathroom or a bedroom with similar furnishings, while other kitchen images showing the dining area are easily mistaken as a bar or a restaurant. In this case, grouping the whole kitchen class with the whole bathroom or restaurant class into a coarse category is suboptimal. Instead, it would be more effective to group confusable images below the category level, such as the images of different classes with similar furnishings as shown in Fig. 2.

Hence, we aim to identify such *confusing clusters* of images in a coarse to fine manner based on high-level appearance. The key idea is to disentangle intra-class variation and inter-class similarity by limiting the intra-class variation within each cluster. With reduced intra-class variation, a specialist model can focus on finding the subtle differences between the categories within the cluster. To this end, we introduce a Hierarchy of Alternating Specialists model, which automatically builds a hierarchical network of specialists based on the unsupervised discovery of confusing clusters. For a given specialist CNN, we find its corresponding confusing cluster by performing clustering in the feature space of its parent model that handles a more general task. This groups images that are visually similar and likely to be confused by the parent model. For assigning images to a model in the hierarchy, we propose a simple routing function which invokes only a small fraction of the models in the whole tree for an input image.

On the other hand, we notice that the spatial layout and the objects in the scene are complementary features for scene categorization (Fig. 4). This seems natural because the scene class is often determined by the way humans use objects in a certain spatial context. For example, the different rooms in a house are typically similar in structure with walls, doors, and windows. However, the objects, such as sofas, books, and dining ware, determine their function as being a living room, office, or dining room. Another notable fact is that the objects do not necessarily stay in the same configuration. To account for this fact, we use two different types of representations in our model: One that is robust to transient local visual elements, and the other that preserves spatial layout. In particular, we propose an alternating architecture, where the architecture of a specialist alternates between the two representations based on its level in the hierarchy. We show that it achieves better performance than the fused features as well as the hierarchical models with a single type of representation.

In summary, our innovations are as follows: (1) We present a hierarchical generalist-specialist model that automatically builds itself based on the unsupervised discovery of confusing clusters in a coarse to fine manner. The confusing clusters allow specialists to focus on subtle differences between images that are visually similar and confusable to their parents. We experimentally validate that our method significantly outperforms baselines including the tree-structured CNNs based on coarse categories. (2) We propose a novel alternating architecture that effectively takes advantage of two complementary representations that capture spatial layouts and transient objects. As minor innovations, we introduce a novel routing function as well as mini-batch soft k-means for end-to-end fine tuning. Beyond the detailed innovations, our proposed algorithm is generalizable to other categorization tasks, and is applicable to any CNN architecture.

2 Related Work

Our method takes the hierarchical mixture of experts approach [7,23], where each expert in the tree structure learns to handle splits of the input space. In light of recent advances in deep neural networks, many researchers have revisited the concept for various recognition tasks [4,21,46,53]. In particular, our method adopts the generalist and specialist model from the work of Hinton et al. [21], which is similar to the mixture of experts, in the sense that each specialist focuses on a confusable subset of the classes, but it has a generalist that takes care of the classes that are not handled by specialists. It also does not require the training of a gating function, allowing models to be trained in parallel. Defining the areas of expertise can be done using a pre-defined semantic hierarchy [11,15], but in this work, we focus on unsupervised approaches [2,37,53,57]. Yan et al. [57] and Murthy et al. [37] use the confusion matrix of a trained classifier to group classes into coarse categories. Ahmed et al. [2] randomly initialize the grouping of classes, then iteratively optimize the grouping as well as the model parameters jointly. In the context of transfer learning, Srivastava and Salakhutdinov [46] take a Bayesian approach to organize the classes into a tree hierarchy.

However, all these approaches partition the input space by grouping categories, while our method partitions the feature space that captures high-level appearance information regardless of class membership, based on the observation that there are visually drastically different sub-classes within each class. This also frees our method from the risk of misclassification in specialties due to severe inter-class similarity and intra-class variation in appearance, from which, the methods using class-based grouping can not recover [2,37,53,57]. Moreover, our method only invokes a limited number of models during testing, which leads to significant computational efficiency gains over existing methods.

In contrast to organizing multiple CNN models, there have been efforts to separate visual features of a single CNN in a tree structure [3,26,31,36,42]. This is especially useful for parallel and distributed learning as demonstrated in Kim et al. [26], where disjoint sets of features, as well as disjoint sets of classes are automatically discovered. In the same spirit of parallelization, but on a much larger scale, Gross et al. [16] deal with a mixture of experts model that does not fit in the memory. Similar to their work, our learned submodels are local in the feature space, and the image-to-model assignment is determined by the distance of the image to the corresponding submodel cluster center.

Numerous work has been done on scene categorization as being one of the fundamental problems in computer vision [17,25,30,38,41,51,55,56,62]. Our work is related to recent attempts to leverage object information within the scene [10,13,14,20,52,63]. However, we do not explicitly detect objects using pre-trained networks or perform rigorous clustering offline to find such visual elements [24,54]. Instead, we let the network capture such information during the end-to-end training process through a network architecture that accounts for objects that can freely move within the scene. Global orderless pooling of convolutional features has a high degree of invariance for encoding local visual elements such as objects. In this way, high-level convolutional filters perform like an object detector [6,60]. Furthermore, we also leverage a global ordered pooling representation which preserves coarse spatial information [35].

3 Method

We first describe our proposed hierarchy of specialists with alternating architecture in Sec. 3.1. We then illustrate how to discover a specialist’s area of expertise in an unsupervised manner in Sec. 3.2. Lastly, we describe the learning objectives as well as the overall training procedure in Sec. 3.3.

3.1 Hierarchy of Alternating Specialists

We propose a hierarchical version of the generalist-specialist models [21], where the child specialist focuses on the task that is more specific than its parents. To achieve this, we begin with a generalist model and then incrementally add specialist models in the next level of the hierarchy, after reaching convergence at the current level. We initialize a new specialist with its parent, or the nearest

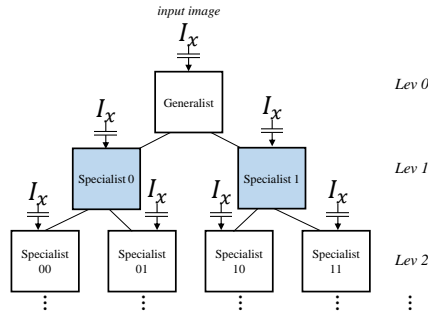


Fig. 3. Hierarchy of alternating specialists. The white and the blue boxes denote network architectures with different global pooling strategy. The assignment of images to models is determined by our routing function, depicted as switches.



Fig. 4. (top) Similar layouts make these scenes confusing, but different objects within them can help determine the correct scene class. (bottom) When scenes are similar in terms of content, their layouts can help distinguish between them.

ancestor that share the network architecture, to inherit its parent’s knowledge as they encode important commonalities of the classes. Note that a specialist model outputs predictions for the same set of categories as the generalist model does. A specialist refines the inherited model towards the finer details to distinguish the classes for images that fall into its specialty. The overall architecture is depicted in Fig. 3. The algorithm stops extending the hierarchy when there is no further improvement, or if the network reaches a pre-specified maximum depth. In this paper, we use a binary tree structure where each parent model has two child models. Every model within this tree shares the low level layers for computational efficiency.

We design this hierarchy of specialists to have an alternating architecture such that specialists at each level have a different model architecture than their parents or children. In particular, we use the global ordered pooling architecture for capturing the rough geometry of the scene, and the global orderless pooling architecture for capturing transient visual elements such as objects. The key idea is that the scene layout and the objects in the scene are complementary for scene classification. Objects can often disambiguate the two images belonging to different categories with similar layouts, while the scene layouts can help distinguish two images, which share the same objects (Fig. 4).

The two architectures differ from each other in how they pool the features in the last convolutional layer before the fully connected layers for the class prediction. First is the global ordered pooling architecture, where the orderless pooling operation (i.e., max- or average-pooling) is performed only within a local spatial window, as in AlexNet [29] and VGG [44]. Thus, the representation preserves the coarse spatial information. The second is the global orderless pooling architecture, in which convolutional features are pooled through global average-pooling, global max-pooling, or VLAD [5], as in NIN [32] and ResNet [19] architecture.

This has a high degree of invariance for encoding local visual elements such as objects, analogous to the widely adopted bag-of-words representation.

Our model uses the original pooling strategy of the base architecture for the generalist at the root node, and alternates between the two architectures for all other elements of our tree structure. To convert one architecture from the other, we either substitute global average pooling with a fully-connected layer (global orderless \rightarrow global ordered), or replace a fully connected layer with global average pooling (global orderless \leftarrow global ordered).

Routing: In order to decide which model in the hierarchy should tackle the input image, we use a simple routing function inspired by the SIFT ratio test [34]. The idea is to let the parent handle the image unless the image has a good membership to any of its children’s area of expertise. We define the routing function to produce a k -dimensional binary vector γ , where the k is the number of children at the current node and $\sum_i \gamma_i \leq 1$. $\gamma_i = 1$ indicates the routing to the i -th child is valid. In the feature space of the current node f_p , given its children’s corresponding *confusing cluster* (Sec. 3.2) centroids μ_k ’s, we compute the distance between the input image I and its nearest centroid μ_i , where $i = \underset{k}{\operatorname{argmin}} \|f_p(I) - \mu_k\|$. We also compute the second nearest centroid μ_j . We then take the ratio of the two distances. If the ratio is less than a threshold τ , the image is assigned to the child node i . Otherwise, the image is assigned to the current node (Eqn. (1)). Then the same routing procedure is performed at the child node i . The decision boundary of the routing function consists of two Apollonius circles where the foci’s are the centroids μ_i and μ_j [4].

$$\gamma_i^{\text{train}}(I) = \begin{cases} 1, & \frac{\|f_p(I) - \mu_i\|}{\|f_p(I) - \mu_j\|} < \tau \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

During testing, we put an additional constraint based on the relative confidence of the prediction between a parent and its child. Intuitively, for those images that are within the specialty of the child ($\gamma_i^{\text{train}}(x) = 1$), we trust the prediction of the child as our answer, when the confidence of the child model is greater than that of its parent model on the given image. Otherwise, we accept the parent model’s prediction and regard the prediction of the child model as unreliable.

$$\gamma_i^{\text{test}}(I) = \begin{cases} 1, & \gamma_i^{\text{train}}(I) \wedge (\operatorname{conf}_i(I) > \operatorname{conf}_p(I)) \\ 0, & \text{otherwise} \end{cases}, \quad (2)$$

where $\operatorname{conf}_i(I) = \max_c P(c|I, \theta_i)$. Since the distance to the clusters is computed in the feature space of the parent models at each level, the total number of models that needs to be invoked is $n_l + 1$ where n_l is the hierarchical level of the selected model ($n_l = 0$ for the generalist). The procedure can also be computed in parallel, at the expense of the number of invoked models (Sec. 4.7).

3.2 Discovering the areas of confusion

We want to partition the input data based on their high-level appearance features, and not by their categorization, thus allowing samples belonging to the

same class to fall into different clusters. Our key insight is that each subset within a class is often associated with different types of inter-class similarity. We perform clustering in the feature space of a parent model to discover *confusing clusters*, the groups of images that are both visually similar and likely to be confused by the parent. This can be interpreted as disentangling intra-class variation and inter-class similarity, as the resulting cluster has limited intra-class variation, and a child model can focus on finding the subtle differences between each categories within the cluster. Also, due to our alternating architecture, we obtain confusing clusters that are both confusing in terms of scene layout and the transient scene objects as we go deeper in the hierarchy.

Feature for clustering: The features from the penultimate layer of a parent model encodes high-level appearance as perceived by the parent. On the other hand, the features from the last fully-connected layer directly encode the class scores by the parent model. The distance of images in these two embedding spaces indicates how likely they are to be distinguished by the parent. In the dataset we tested, the combination of these embeddings produced a marginally better results compared to using each of them separately. In the experiments, we report the result using the combined features, unless otherwise specified.

Incremental hard clustering: As described in Sec. 3.1, we build our hierarchical model in an incremental manner, where the models in the next hierarchical level are added when their parent models have converged. As such, we discover confusing clusters by performing hard k -means clustering on the features of a converged parent model. Once initialized with these clusters, we can further fine-tune them end-to-end using the soft k -means layer described below.

Soft k -means layer for fine-tuning: We propose to use mini-batch-based soft k -means that allows end-to-end fine-tuning. For each model θ , we update the centroids μ_k through back-propagation to optimize the following objective function:

$$L_{clust}(\theta, \mu; I_i) = \sum_{k=1}^K \sum_{i=1}^N w_{ik} \|f_{\theta}(I_i) - \mu_k\|^2, \quad (3)$$

where

$$w_{ik} = \frac{e^{-m\|f_{\theta}(I_i) - \mu_k\|^2}}{\sum_{k=1}^K e^{-m\|f_{\theta}(I_i) - \mu_k\|^2}}, \quad (4)$$

and $f_{\theta}(I_i)$ denotes an image representation in the mini-batch. The parameter m decides the softness of the membership w_{ik} of x_i belonging to cluster k . We set m to $1/(8\sigma^2)$ where σ is the average of the standard deviation to the cluster center, which is computed during the hard k -means clustering.

3.3 Training

Classification Loss: As we allow the samples belonging to the same class to be in different clusters, it may introduce class imbalance in the training set of the specialists. Thus, we weigh the cross-entropy loss with the inverted document frequency similar to [33]. This better accounts for under-represented classes

within the cluster. We computed the inverted document frequency as a running average to allow changes caused by clustering.

$$L_{class}(\theta; x) = - \sum_c \left(\log \frac{N}{n_c} \right) \log(P[c|x, \theta]) \quad (5)$$

Training Objective: Our final training objective consists of clustering loss and classification loss as follows:

$$L_{total}(\theta; x) = \sum_{d \in \mathcal{D}} L_{class}(\theta_d; x) + \sum_{d \in \mathcal{D}, d \notin \mathcal{L}} L_{clust}(\theta_d; x). \quad (6)$$

\mathcal{D} and \mathcal{L} denote the sets of all nodes and leaf nodes in the hierarchy, respectively.

Implementation Details: The parameters of the shared low-level layers and the layers of the parents are kept frozen until the fine-tuning stage of the overall network. As the architecture of the specialist model alternates between the levels in the hierarchy, a specialist is initialized with its grandparent model whom it shares the architecture with. We initialized our base models with the models pre-trained on ImageNet, then fine-tuned them on the target dataset, with an exception in the experiment on CIFAR-100, where the base model is trained from the scratch until its accuracy reached the performance for the same model reported in [3,57]. The number of confusing clusters K are set to 2. The threshold τ for the routing function is empirically selected as 0.96. We used stochastic gradient descent for the optimization. The deployed learning rate was 0.001, and was reduced by a factor of 10 when the loss plateaus. To combat overfitting, data augmentation techniques such as random cropping, scaling, aspect ratio setting [47], and color jittering [49] were applied. We used an image resolution of 224×224 . Our models are implemented using PyTorch [39].

4 Experiments

We perform quantitative comparison to evaluate our approach and its components (Sec. 4.2-4.3). For a direct comparison with other tree-structured networks, we show the results of our method on a general image classification task (Sec. 4.4). We then show how regions of interest are changed in the specialists as compared to that of the generalist (Sec. 4.5). We also visualize the learned hierarchy, which qualitatively validates our premises on feature-based grouping (Sec. 4.6).

4.1 Datasets and evaluation methodology

Dataset: We performed experiments on the widely used SUN database [55]. The original number of scene categories in this dataset is 397. However, the majority of categories contain just around 100 example images. To alleviate the potential overfitting problem, we create a subset of SUN-397 [55], the (1) *SUN-190 dataset*, which consist of classes that contains at least 200 examples, resulting in 48K images in total. Following Agrawal et al. [1], we randomly divide the data

for training, test, and validation with the proportion of 60%, 30%, and 10%. We used this dataset for comprehensive study, as its size allows us to evaluate a variety of design choices. We also performed experiments on another publicly available dataset, the (2) *Places-205 dataset* [63], which contains 2.5M images. For the Places-205 dataset, we treated the validation set as our test set. Finally, for the comparison with the existing tree-structured networks, we also report our results on the (3) *CIFAR-100 dataset* [28], a standard image classification benchmark which contains 60K images in total.

Evaluation Metric: Following the standard protocol [1,63], we report one-vs.-all classification accuracy averaged over all classes. We report both top-1 accuracy and top-5 accuracy for SUN-190 and Places-205 [63], and top-1 for the CIFAR-100 dataset [28]. In all our experiments, test images for evaluation were resized to a resolution of 224×224 and we perform single-view testing, i.e., no averaging of multiple crops [1,29,63] were performed.

Base model: On the SUN-190 and Places-205 [63] datasets, we used AlexNet* [27] as our base model, which is a slimmer version of the original AlexNet [29]. We let the specialists share the parameters with the lower layers of the generalist up to conv4. For the global ordered representation, we use the AlexNet* architecture as is. For the global orderless representation, we keep the layers of AlexNet* up to conv5 and add a conv6 layer with $768 \ 3 \times 3$ filters, with a global average pooling layer between conv6 and fc7. On CIFAR-100 [28] dataset, NIN-C100 [32] is used as our base model. It is used as is for the global orderless representation. For the global ordered representation, the global average pooling layer was replaced with two fully-connected layers, each with 1024 and 100 dimensional output.

4.2 Scene classification results

In order to evaluate our premise that specialists trained on *confusing clusters* are better than those trained on coarse categories, we compare with a network of experts based on coarse categories. In particular, we compare a two-level hierarchical model similar to HD-CNN [57], but with AlexNet* [27] (HD-CNN*) as a baseline for a fair comparison with our method. For this baseline method, spectral clustering was performed on the covariance matrix of class predictions of the generalist model for discovering the groups of confusing categories as in [21]. The final prediction is made using the weighted average of predictions as in [57]. We experimented with a different number of clusters of 2, 4, and 8 for this model. Furthermore, we compare our approach with a simple ensemble model, where the models are trained with different initializations and the predictions are averaged. We also report the performance of the fine-tuned single AlexNet* [27] model, which is also our generalist model at the root of the hierarchy.

In Table 1, we compare our performance with the aforementioned baselines on the SUN-190 dataset. All of our models outperform the baselines, where our best model with a 3-level hierarchy achieved a classification accuracy of 66.41% for the Top-1 prediction, exceeding the accuracy of the coarse-category-based model (HD-CNN*) by 2.76%. The performance of our proposed model consistently improves as we increase the number of levels in the hierarchy. In contrast,

Table 1. Scene classification accuracy on the Sun-190 dataset. All compared models are based on AlexNet* [27]. The statistics are based on single-view testing. (Lev: the hierarchical levels, K: the number of clusters, N: the number of ensembles)

Method		Top-1	Top-5	Method		Top-1	Top-5
Proposed	Lev 1	66.13	89.66	AlexNet*[27](Ordered)	Lev 0	63.46	89.18
	Lev 2	66.37	89.85	AlexNet*-Orderless	Lev 0	61.79	88.14
	Lev 3	66.41	89.96	Fusion	Lev 0	64.45	89.36
HD-CNN* [57]	K = 2	63.11	88.81	Model 1:	Lev 1	64.02	89.27
	K = 4	63.62	87.64	global ordered	Lev 2	64.33	89.44
	K = 8	63.65	84.08	pooling only	Lev 3	64.43	89.48
Simple Ensembles	N = 2	64.19	89.47	Model 2:	Lev 1	62.71	88.54
	N = 4	64.66	89.72	global orderless	Lev 2	63.14	88.76
	N = 8	64.99	89.96	pooling only	Lev 3	63.08	88.69

HD-CNN* only has marginal improvements in the Top-1 accuracy, while the Top-5 accuracy drops as the number of clusters increases. This demonstrates the effectiveness of our model in discovering the correct hierarchical organization of image data while overcoming the intra-class variation issues inherent in conventional tree-structured models. We also observe that while our model achieves well-balanced clusters, the spectral clustering resulted in high bias in the number of classes per coarse category. The simple ensembles is also inferior to our approach which outputs the prediction of a single specialist model.

We also show the scene classification performance on the Places-205 [63] dataset on Table 2. Our approach provides an improvement of 2.87% over the base model at Top-1 accuracy. Similarly as in SUN-190, we observe that the accuracy of the proposed model increases as we increment the number of levels in the hierarchy.

4.3 Benefits of Alternating Architecture

The performance of architectures with global ordered pooling (AlexNet* [27]) and global orderless pooling (AlexNet*-Orderless) are shown in Table 1 and Table 2, for the SUN-190 and Places-205 datasets, respectively. Both models achieve similar accuracy, while global ordered pooling shows slightly better performance. Meanwhile, the IoU of the correct prediction was 78.1% (with the overall prediction overlap ratio being and 73.2%). This quantitatively validates our assumption that the two representations are complementary. We also evaluated the performance of fused features, one with early fusion that concatenates two representations before the last fully connected layer, and the other with late fusion where the predictions of the two architectures are averaged. The early fusion did not yield competitive classification accuracy. On the other hand, the late fusion (Fusion in Table 1 and 2) achieves better performance than using each representation separately, however, does not reach the classification accuracy of our proposed alternating architecture.

Table 2. Scene classification accuracy on the Places-205 [63] dataset using single-view testing. All models are based on AlexNet* [27].

Method		Top-1	Top-5
AlexNet*[27](Ordered)	Lev 0	48.67	79.24
Proposed: alternating architecture	Lev 1	50.21	79.82
	Lev 2	51.42	80.67
	Lev 3	51.54	80.76
Model 1: global ordered pooling only	Lev 1	49.99	80.09
	Lev 2	50.21	80.30
	Lev 3	50.28	80.26
AlexNet*-Orderless	Lev 0	48.19	78.23
Model 2: global orderless pooling only	Lev 1	48.99	79.01
	Lev 2	49.31	79.48
	Lev 3	49.25	79.49
Fusion	Lev 0	49.19	79.54

Table 3. Image classification accuracy on CIFAR-100 with single-view testing. All models are based on NIN-C100 [32].

Method		Top-1
NIN-C100[32](Orderless)	Lev 0	64.73
Proposed: alternating architecture	Lev 1	67.32
	Lev 2	67.61
	Lev 3	67.70
Model A: global orderless pooling only	Lev 1	66.92
	Lev 2	66.70
	Lev 3	66.62
NIN-C100-Ordered	Lev 0	64.67
Model B: global ordered pooling only	Lev 1	65.64
	Lev 2	65.74
	Lev 3	65.48
Fusion	Lev 0	66.83

Table 4. Comparison with other tree-structured models on CIFAR-100 [28]. All models are based on NIN-C100 [32]. The accuracies are based on single-view testing.

Method	hierarchy levels	#model choices	#model selected	#models invoked	Accuracy (%)
NIN-C100 [32]	0	1	1	1	64.73
Proposed	1	3	1	1-2	67.32
	2	7	1	1-3	67.61
	3	15	1	1-4	67.70
HD-CNN (best) [57]	1	9	9	10	65.64
NoFE [2]	1	10	1	2	65.91
BranchConnect [3]	1	10	1	10	66.10
			5	10	66.45

Furthermore, in Table 1 and 2, we compare with other versions of our method—hierarchy of specialists *without* the alternating architecture, that is, using only a single type of representation. In particular, we report the results of Model 1 that uses global ordered pooling architecture, and Model 2 with global orderless pooling architecture. Both models were trained with the same training protocol as our proposed model. While the performance of our proposed model with its alternating architecture improves with an increasing depth of the hierarchy, models with a non-alternating architecture have marginal or no observable performance gain. We suspect that this is due to the fact that our alternating architecture is better at yielding confusing clusters, in terms of both coarse spatial layout and the objects in the scene, by using two different types of feature sets.

To show the observation holds for other networks, we repeated the same experiments using the NIN-C100 [32] architecture on CIFAR-100 [28] in Table 3. Unlike AlexNet* [27], which has a global ordered pooling architecture, NIN-

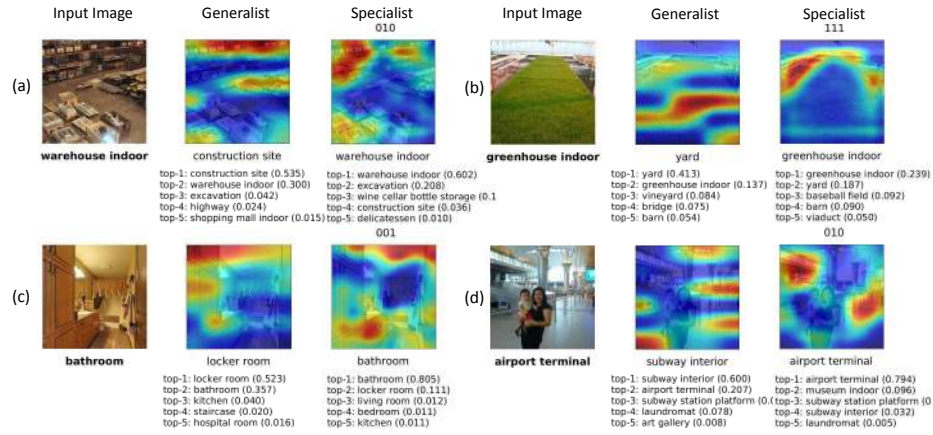


Fig. 5. (left) Input images and ground-truth category. The top-5 predictions and the visualization of class activation maps (CAM) of the top predicted class for the generalist (center) and the selected specialist (right). (See supplementary for more results.)

C100 [32] has a global-orderless-pooling architecture by default. We observe that our alternating architecture clearly outperforms other strategies. The Model A and B denote the hierarchy of specialists with a single type of representation, using global orderless- and global ordered- pooling, respectively.

4.4 Comparison with existing tree-structured CNNs on CIFAR-100

For a direct comparison with other tree-structured networks, we show the results of our architecture on the image classification task of the CIFAR-100 [28] dataset in Table 4. We compare with HD-CNN [57], NofE [2], and BranchConnect [3]. All these methods train their experts on coarse categories (class-based grouping) while our method alone uses *confusing clusters*. Furthermore, they require additional networks or layers to be used for gating. We show the recalls reported in their original paper, except for NofE [2] in which we used the recalls reported in [3] in order to match the performance of the base model for a fair comparison. All models are based on the NIN-C100 [32] architecture. We also illustrate the number of models to choose from, the number of selected models, and the total number of invoked models on the same table. Our approach outperforms all other methods despite the fact that it outputs the prediction of a single specialist network, rather than averaging predictions of multiple networks. It also invokes the least number of models. In particular, our method outperforms the best baseline BranchConnect [3] with significantly fewer models invoked.

4.5 Comparison of Regions of Interest (ROI)

The benefit of our proposed architecture lies in the specialists' ability to discriminate between classes based on subtle details for images that falls into their

specialty. As specialists are trained on the subset of data which reflects their specialty, it evolves to focus on such details to better accommodate the classification task at hand. To illustrate these changes in activation patterns, we investigated how the regions of interest (ROI) of the specialist models differ from those of the generalist models. We visualize the corresponding class activation maps (CAM) [43,61] for the specialists and the generalists. Since CAMs show the regions that contributed to the prediction of the class in question, we are able to tell which regions in the image contributed to the correct (or the incorrect) prediction.

Fig. 5 shows the CAMs of the top predicted class for both the generalist and the specialists. We only show examples where the specialists with the depicted results are invoked by our routing function. We observe that the specialists are good at focusing on fine-grain details as compared to the generalist. For example, in Fig. 5 (a), the generalist reasonably predicted the scene category as construction site, based on the construction materials on the right side of the image. However, the specialist was able to focus more on the boxes, predicting the correct scene class of warehouse indoor. In Fig. 5 (b), generalist predicted yard for the scene class, based on the grass field in the center of the image. However, the specialist paid more attention to plants and frames on the sides to predict the correct class of greenhouse indoors.

4.6 Visualization of Learned Hierarchy of Specialties

We visualize the learned hierarchy of images in Fig. 6. For each centroid of the discovered confusing clusters that the specialists were trained on, we depict the top 10 nearest neighbor images in the feature space for SUN-190. We observe that each cluster consists of visually coherent and easily confusable images from different scene classes. At the same time, different instances of the same class appear in multiple clusters that are visually distinct. For example, a subset of the kitchen images, which are visually similar to bathrooms with base cabinets, appear in the cluster of the Specialist 001, while the subset of the same category that look similar to restaurants and bars are found in the cluster of the Specialist 10. This visualization strongly supports our underlying idea of *confusing clusters*.

4.7 Computational Time

Our model can be run in parallel or sequentially. Running sequentially minimizes the number of invoked models, thus saving memory at the expense of time. The opposite is true when running in parallel. Let $t_A = t_l + t_u$ be the execution time for the base model, where t_u and t_l denote the time spent on the upper layers and the shared lower layers. Let t_r be the time spent for routing and L the hierarchical levels. When run sequentially, the best case is $t_A + t_r$ when routed to the generalist, while the worst is $t_l + L \cdot (t_u + t_r)$ when routed to a leaf specialist. On an NVIDIA GTX1080Ti with batch size 512 using AlexNet*, it takes 105, 121, and 138ms for our models with $L = 1, 2, 3$, respectively. AlexNet* takes 87ms. When fully parallelized, each model is run in parallel, then a model is selected, which takes $t_A + t_r$. It takes 89ms for all our models ($L = 1, 2, 3$).



Fig. 6. Visualization of the learned hierarchy on the SUN-190 dataset. A three level hierarchy is shown, with the 10 top images associated with each specialist.

5 Conclusion

We introduced a novel hierarchy of alternating specialists for tackling inter-class similarity and intra-class variation in scene categories. The global feature pooling strategy of the specialist model alternates at each level to account for both coarse scene layout and transient objects, which are both essential for accurate scene classification. For defining the area of expertise for each child model, we discover confusing image clusters by performing clustering based on the learned features of the parent model, thereby obtaining image clusters that are visually coherent and confusing at the same time. Our method invokes only a small fraction of the models in the whole tree for an input image. We experimentally show that our method achieves significant improvement over baselines including existing tree-structured models that use class-based grouping. Our algorithm is applicable to a variety of CNN models and visual category recognition tasks.

Acknowledgment Partially supported by NSF grant No. CNS-1405847.

References

1. Agrawal, P., Girshick, R., Malik, J.: Analyzing the performance of multilayer neural networks for object recognition. In: ECCV (2014)
2. Ahmed, K., Baig, M.H., Torresani, L.: Network of experts for large-scale image categorization. In: ECCV (2016)
3. Ahmed, K., Torresani, L.: Branchconnect: Large-scale visual recognition with learned branch connections. In: WACV (2018)
4. Aljundi, R., Chakravarty, P., Tuytelaars, T.: Expert gate: Lifelong learning with a network of experts. In: CVPR (2017)
5. Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: CVPR (2016)
6. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: Quantifying interpretability of deep visual representations. In: CVPR (2017)
7. Bishop, C.M., Svenskn, M.: Bayesian hierarchical mixtures of experts. In: Uncertainty in Artificial Intelligence (2002)
8. Chen, W., Chen, X., Zhang, J., Huang, K.: Beyond triplet loss: a deep quadruplet network for person re-identification. In: CVPR (2017)
9. Cheng, G., Zhou, P., Han, J.: RIFD-CNN: Rotation-invariant and fisher discriminative convolutional neural networks for object detection. In: CVPR (2016)
10. Cheng, X., Lu, J., Feng, J., Yuan, B., Zhou, J.: Scene recognition with objectness. Pattern Recognition (2018)
11. Deng, J., Berg, A.C., Fei-Fei, L.: Hierarchical semantic indexing for large scale image retrieval. In: CVPR (2011)
12. Deng, J., Ding, N., Jia, Y., Frome, A., Murphy, K., Bengio, S., Li, Y., Neven, H., Adam, H.: Large-scale object classification using label relation graphs. In: ECCV (2014)
13. Dixit, M., Chen, S., Gao, D., Rasiwasia, N., Vasconcelos, N.: Scene classification with semantic fisher vectors. In: CVPR (2015)
14. Dixit, M.D., Vasconcelos, N.: Object based scene representations using fisher scores of local subspace projections. In: NIPS (2016)
15. Goo, W., Kim, J., Kim, G., Hwang, S.J.: Taxonomy-regularized semantic deep convolutional neural networks. In: ECCV (2016)
16. Gross, S., Ranzato, M., Szlam, A.: Hard mixtures of experts for large scale weakly supervised vision. In: CVPR (2017)
17. Guo, S., Huang, W., Wang, L., Qiao, Y.: Locally supervised deep hybrid model for scene recognition. TIP (2017)
18. Guo, Y., Zhao, G., Pietikäinen, M., Xu, Z.: Descriptor learning based on fisher separation criterion for texture classification. In: ACCV (2010)
19. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
20. Herranz, L., Jiang, S., Li, X.: Scene recognition with cnns: objects, scales and dataset bias. In: CVPR (2016)
21. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. Arxiv preprint arXiv:1503.02531 (2015)
22. Hwang, S.J., Sigal, L.: A unified semantic embedding: Relating taxonomies and attributes. In: NIPS (2014)
23. Jordan, M.I., Jacobs, R.A.: Hierarchical mixtures of experts and the em algorithm. Neural Computation (1994)

24. Juneja, M., Vedaldi, A., Jawahar, C., Zisserman, A.: Blocks that shout: Distinctive parts for scene classification. In: CVPR (2013)
25. Khan, S.H., Hayat, M., Porikli, F.: Scene categorization with spectral features. In: CVPR (2017)
26. Kim, J., Park, Y., Kim, G., Hwang, S.J.: Splitnet: Learning to semantically split deep networks for parameter reduction and model parallelization. In: ICML (2017)
27. Krizhevsky, A.: One weird trick for parallelizing convolutional neural networks. arXiv preprint arXiv:1404.5997 (2014)
28. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images (2009)
29. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
30. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: CVPR (2006)
31. Li, F., Neverova, N., Wolf, C., Taylor, G.: Modout: Learning multi-modal architectures by stochastic regularization. In: FG (2017)
32. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint arXiv:1312.4400 (2013)
33. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV (2017)
34. Lowe, D.G.: Object recognition from local scale-invariant features. In: ICCV (1999)
35. Mousavian, A., Kosecka, J.: Deep convolutional features for image based retrieval and scene categorization. arXiv preprint arXiv:1509.06033 (2015)
36. Murdock, C., Li, Z., Zhou, H., Duerig, T.: Blockout: Dynamic model selection for hierarchical deep networks. In: CVPR (2016)
37. Murthy, V.N., Singh, V., Chen, T., Manmatha, R., Comaniciu, D.: Deep decision network for multi-class image classification. In: CVPR (2016)
38. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. IJCV (2001)
39. Paszke, A., Gross, S., Chintala, S., Chanan, G.: Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration. <http://pytorch.org/> (2017)
40. Qian, Q., Jin, R., Zhu, S., Lin, Y.: Fine-grained visual categorization via multi-stage metric learning. In: CVPR (2015)
41. Quattoni, A., Torralba, A.: Recognizing indoor scenes. In: CVPR (2009)
42. Sabour, S., Frosst, N., Hinton, G.E.: Dynamic routing between capsules. In: NIPS (2017)
43. Selvaraju, R.R., Das, A., Vedantam, R., Cogswell, M., Parikh, D., Batra, D.: Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. In: ICCV (2017)
44. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
45. Somanath, G., Kambhamettu, C.: Abstraction and generalization of 3d structure for recognition in large intra-class variation. In: ACCV (2010)
46. Srivastava, N., Salakhutdinov, R.R.: Discriminative transfer learning with tree-based priors. In: NIPS (2013)
47. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR (2015)
48. Tu, Z.: Probabilistic boosting-tree: Learning discriminative models for classification, recognition, and clustering. In: CVPR (2005)

49. Urban, G., Geras, K.J., Kahou, S.E., Aslan, O., Wang, S., Caruana, R., Mohamed, A., Philipose, M., Richardson, M.: Do deep convolutional nets really need to be deep and convolutional? arXiv preprint arXiv:1603.05691 (2016)
50. Wang, J., Liu, Z., Wu, Y., Yuan, J.: Mining actionlet ensemble for action recognition with depth cameras. In: CVPR (2012)
51. Wang, L., Guo, S., Huang, W., Xiong, Y., Qiao, Y.: Knowledge guided disambiguation for large-scale scene classification with multi-resolution cnns. TIP (2017)
52. Wang, Z., Wang, L., Wang, Y., Zhang, B., Qiao, Y.: Weakly supervised patchnets: Describing and aggregating local patches for scene recognition. TIP (2017)
53. Warde-Farley, D., Rabinovich, A., Anguelov, D.: Self-informed neural network structure learning. arXiv preprint arXiv:1412.6563 (2014)
54. Wu, R., Wang, B., Wang, W., Yu, Y.: Harvesting discriminative meta objects with deep cnn features for scene classification. In: ICCV (2015)
55. Xiao, J., Hays, J., Ehinger, K.A., Oliva, A., Torralba, A.: Sun database: Large-scale scene recognition from abbey to zoo. In: CVPR (2010)
56. Xiao, Y., Wu, J., Yuan, J.: mcentrist: A multi-channel feature generation mechanism for scene categorization. TIP (2014)
57. Yan, Z., Zhang, H., Piramuthu, R., Jagadeesh, V., DeCoste, D., Di, W., Yu, Y.: HD-CNN: Hierarchical deep convolutional neural networks for large scale visual recognition. In: ICCV (2015)
58. Zhao, B., Li, F., Xing, E.P.: Large-scale category structure aware image categorization. In: NIPS (2011)
59. Zheng, W.S., Gong, S., Xiang, T.: Person re-identification by probabilistic relative distance comparison. In: CVPR (2011)
60. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Object detectors emerge in deep scene cnns. In: ICLR (2014)
61. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: CVPR (2016)
62. Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. PAMI (2017)
63. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., Oliva, A.: Learning deep features for scene recognition using places database. In: NIPS (2014)