

A TRAINING DETAILS

A.1 PRETRAINING PRE-TRAINED MODELS

Architectures: The ResNet architectures used is *ResNetV2*, with batch norm replaced by group norm and weight standardisation used in the convolutions, as in Kolesnikov et al. (2019). We consider 18, 34, 50 and 101 layer models.

Datasets: As mentioned, we use JFT-300M and ImageNet21k as our pretraining datasets. We split these datasets into subsets following the protocol set in Puigcerver et al. (2020), where a generalist model is finetuned for 2 epochs on each subset of the pretraining dataset. We repeat this for 18, 34 and 101 layer ResNets on JFT.

Training: For the vast majority of pre-trained models in this work, models were pre-trained by taking a generalist model and fine-tuning it for 2 epochs on a subset of the pretraining data. We also consider training architectures from scratch on subsets of the pre-training data; these generally performed slightly worse, but are included in the ‘All Experts’ pool. We pre-train models with the same settings described in Puigcerver et al. (2020).

A.1.1 VTAB: FINETUNING PRE-TRAINED MODELS

For both generalist and expert models, pre-trained models are trained on a target dataset by adding a new dense layer at the head, with units equal to the number of classes of the output space. All models were trained using SGD with momentum, with the momentum parameter set to 0.99. They were trained on Google Cloud TPUv3s with batch size 512.

Preprocessing

We use the task-specific preprocessing suggested by BiT-HyperRule (Kolesnikov et al., 2019); this dictates resolutions (pre and post crop) and whether to flip left/right. Except for AugEnsembles, we do not use MixUp as they do. For all methods, we train 3 models on each task in order to give rough confidence intervals. As we explore training large numbers of models, it was not computationally feasible to train more.

Default Hyperparameter Sweep

Unless hyperparameters are sampled as in Appendix A.2, models undergo a default sweep of parameters on the target dataset. For ExpertEnsembles, this means each feature extractor was fine-tuned 4 times on the downstream dataset. For AugEnsembles, it means for a given randomly sampled augmentation setting, the generalist model was fine-tuned 4 times on the downstream dataset. This default sweep is the product of two learning rates (0.1 and 0.01) and two schedule lengths (2500 and 10000 steps). For the schedule, we use an exponential step decay schedule; it warms-up linearly for 20% of the schedule, then decays by a factor of 0.1 at each subsequent 20% interval.

A.2 TRAINING HYPERENSEMBLES

As discussed, our hyper ensembles start with a single pre-trained generalist model - in this case, either a BiT-L or BiT-M ResNet-50x1, which were pretrained on JFT and ImageNet21k respectively. We use the random hyperparameter search space defined in Kolesnikov et al. (2019), which we recount here for convenience:

- Weight decay to init $\sim \text{LogUniform}(1 \times 10^{-6}, 1 \times 10^{-1})$
- Dropout rate $\sim \text{Uniform}(0.0, 0.5)$
- Learning rate $\sim \text{LogUniform}(1 \times 10^{-4}, 1 \times 10^{-1})$
- Schedule length $\sim \text{Choice}(500, 1k, 2k, 5k, 8k, 16k)$

A.3 TRAINING AUGENSEMBLES

Similar to the hyper-ensembles, the AugEnsembles start with a single pre-trained generalist model. For all augmentations, the final resolution for each dataset is that defined by the BiT-HyperRule in (Kolesnikov et al., 2019), and we rescale intensities to have a range of -1.0 to 1.0. Note that augmentations themselves are random from image to image, e.g. the color distortion augmentations

will apply different magnitudes of distortions (hence why for a fixed augmentation setting, there is still benefit to diversity). We sample whether or not to include these distortions and the parameters of these distortions themselves (e.g. distortion level). We randomly sample augmentations as so:

- Choose between inception crop and random crop
 - If random crop: Sample initial resolution $\sim \text{Uniform}(1.05, 1.30) \times$ the final resolution
- If flipping left-right is mandated in the BiT-HyperRule, choose to flip with probability 50%.
- (For non artificial datasets), apply each of the following color distortions with probability 50%:
 - *Random additive brightness modulation.*
Adds to all channels some $\delta \sim \text{Uniform}(-\delta_{\max}, \delta_{\max})$.
We sampled $\delta_{\max} \sim \text{Uniform}(0.01, 0.2)$
 - *Random additive hue modulation.*
Adds to the Hue value in HSV space some value $\delta \sim \text{Uniform}(-\delta_{\max}, \delta_{\max})$.
We sampled $\delta_{\max} \sim \text{Uniform}(0.01, 0.2)$
 - *Random multiplicative saturation modulation.*
Multiplies Saturation channel in HSV by a factor $s \sim \text{Uniform}(s_{\min}, s_{\max})$.
We sampled $s_{\min} \sim \text{Uniform}(0.25, 0.75)$, $s_{\max} \sim \text{Uniform}(2 \times s_{\min}, 4 \times s_{\min})$
 - *Random multiplicative contrast modulation.*
Multiplies per-channel standard deviation by a factor $s \sim \text{Uniform}(s_{\min}, s_{\max})$.
We sampled $s_{\min} \sim \text{Uniform}(0.25, 0.75)$, $s_{\max} \sim \text{Uniform}(2 \times s_{\min}, 4 \times s_{\min})$
- Apply mixup with probability 50%
 - If mixup is applied, sample α parameter $\sim \text{Uniform}(0.01, 0.2)$

Note that we did not apply color distortion to datasets which were artificially generated, e.g. dSprites, as that results in significantly worse performance. That being said, it is not clear what augmentations to use for such datasets. We found the greatest gains from this method to be on specialised datasets. We suspect this is because the default augmentations specified by the BiT HyperRule are not well suited for such datasets; in particular, we found PatchCamelyon benefited the most from different augmentations.

For both augmentation ensembles and hyper ensembles, we checked that the models finetuned on each dataset had a roughly similar distribution of validation accuracies as expert models in order to ascertain these were fair ways of generating models.

A.4 IMAGENET: TRAINING MODELS

Data: We use 96% of the train data for training individual models and the remaining 4% for constructing ensembles. We report numbers on the official validation split.

Preprocessing: During training we resize to 512×512 pixel images, then randomly crop to 480×480 , and also randomly flip images horizontally. At evaluation time we simply resize images to 480×480 pixel. We train five random trials of each approach.

Training: We train on Cloud TPUv3s with a batch size of 512, using SGD with Momentum as for VTAB models, with momentum parameter set to 0.99. For the ExpertEnsembles, we use the two learning rates from the default hyperparameter sweep above (0.1 and 0.01), but now use two longer schedules (10k and 20k steps). For the HyperEnsembles, we use the same hyperparameter search space defined in Appendix A.2, but we explore longer schedules of length 10k, 15k, 20k, 25k and 30k steps.

B EVALUATION DETAILS

B.1 THE VISUAL TASK ADAPTATION BENCHMARK (VTAB)

The Visual Task Adaptation Benchmark consists of 19 tasks. They are split into three categories:

- **Natural tasks** CalTech101 (Li et al., 2004) · CIFAR100 (Krizhevsky, 2009) · Street View House Numbers (SVHN - Netzer et al. (2011)) · Describable Textures (DTD - Cimpoi et al. (2014)) · Oxford Flowers (Nilsback & Zisserman, 2006) · Oxford Pets (Parkhi et al., 2012) These tasks contain ‘classical’ natural real-world images obtained with a camera.
- **Specialised tasks** EuroSAT (Helber et al., 2019) · Diabetic Retinopathy (Kaggle & EyePacs, 2015) PatchCamelyon (Veeling et al., 2018) · Remote Sensing Image Scene Classification (RESISC - Cheng et al. (2017)) These are datasets of arguably ‘natural’ images which were captured with specialised photographic equipment.
- **Structured datasets** DeepMind Lab (Object distance prediction - Zhai et al. (2019)) · SmallNOB (Azimuth & Elevation prediction - LeCun et al. (2004) CLEVR (Counting & Distance prediction Johnson et al. (2017) · KITTI (Vehicle distance prediction Geiger et al. (2012)) · dSprites (pixel location & orientation prediction - Matthey et al. (2017)) These assess understanding of scene structure in some way, predominately from synthetic environments. Example tasks include 3D depth estimation and counting.

Validation Performance.

In order to facilitate in-depth study without over-evaluation on the test data, for some plots in the appendix we explore ensemble behaviour on validation data. In this context, given candidate models which have been finetuned on the 800 datapoints, we require data to a) make ensembling decisions (e.g. for the greedy algorithm) and b) evaluate ensemble performance. We partition the 200 validation points into an 80/20 split, using 160 points to make ensembling decisions and 40 for evaluation. We use k -fold cross validation with $k = 15$, comparing final mean accuracy across iterations and datasets. We generally found that this was representative; i.e. ranking of models according to this setup was the same as for evaluation on the test set.

B.2 IMAGENET ROBUSTNESS VARIANTS

Given ensembles trained on ImageNet, we assess models using the suite of ImageNet variants collected in Djolonga et al. (2020). Chiefly speaking, these variants induce some kind of distribution shift on the input images while still retaining the same label space.

- **ImageNet Corrupted** (*ImageNet-C* - (Hendrycks & Dietterich, 2019)) The original ImageNet images, with artificial corruptions such as blur and snow applied.
- **Re-collected data** These datasets consist of data that was re-collected and labelled with ImageNet labels. ImageNet trained classifiers usually experience an accuracy decrease on these datasets.
 - ImageNet Adversarial (*ImageNet-A* - (Hendrycks et al., 2019)) Collection of new data which ResNet-50 models failed to classify.
 - ImageNet-R (Hendrycks et al., 2020) Collection of cartoons/art/tatoos/toys etc with original ImageNet labels, which ResNet-50 models failed to classify.
 - ImageNet-v2 (Recht et al., 2019) Data recollected in a method that closely mimics the original protocol as best as possible.
 - ObjectNet (Barbu et al., 2019) Collection of data with controls for backgrounds, rotation and imaging viewpoint.
- **Video datasets** These datasets consist of videos where the frames are labelled with ImageNet classes. In these dataset, as well as the accuracy of the frames, the $pm-k$ metric is introduced, whereby given an anchor frame, a classifier is only considered correct if it classifies the surrounding $2k + 1$ frames correctly.
 - **YouTube Bounding Boxes** (YTBB - (Real et al., 2017; Shankar et al., 2019))
 - **ImageNet Vidrobust** (Deng et al., 2009; Shankar et al., 2019)

C ABLATIONS AND EXPERIMENTAL ANALYSIS

C.1 FROM-SCRATCH APPROACHES ARE NOT COMPETITIVE IN THE LOW-DATA REGIME

All across the transfer-learning literature there is clear evidence that as the number of available samples for training decreases, the less competitive approaches based on training from scratch become. To demonstrate that also happens when using ensembles we constructed *HyperEnsembles* starting from a random-initialization and from generalist pre-trained model and evaluate them in VTAK-1K. Figure 5 shows the result - training from scratch is hopeless. Even while ensembling many models, it can't compete with a single fine-tuned strong generalist.

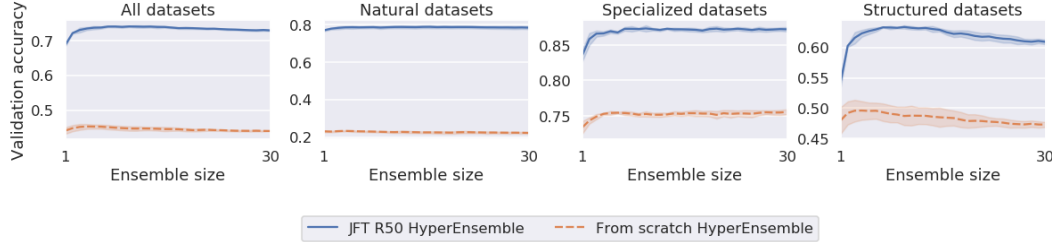


Figure 5: VTAK validation accuracy of *HyperEnsembles* trained from scratch and *HyperEnsembles* trained from a generalist pre-trained model (JFT-R50).

C.2 NUMBER OF MODELS BEING PICKED

Due to the greedy algorithm, different numbers of models are chosen for different datasets. We show in Table 4 the number of experts chosen for each; in particular we note that on structured datasets, where there are less relevant models from pre-training, it tends to use a large number of models. We note also that typically there is no significant difference in the number of models being picked by different methods.

C.3 RANDOM VS FILTERING PRE-TRAINED MODELS

It is possible that the use of experts, or generalists, is simply the transfer learning equivalent of random initialisation - i.e. that diversity in initialisation for the sake of diversity in initialisation is

Table 4: Median number of experts chosen by each method for each dataset. All are pre-trained on JFT, and all are based on ResNet-50 architectures unless otherwise specified

	mean	caltech101	cifar100	dtd	flowers	pets	sun397	svhn	camelyon	eurosat	resisc45	retino	clever.count	clever.closest	dmlab	dsprites.xpos	dsprites.orient	kitti	smallnorb.azmth	smallnorb.elev
		Natural							Specialised				Structured							
D SeedEnsemble	9.4	5	11	15	2	3	4	10	10	8	13	15	4	15	15	3	15	10	11	15
D AugEnsemble	8.2	3	7	13	2	8	4	7	8	5	6	10	3	7	15	10	11	8	15	14
D HyperEnsemble	9.1	10	13	12	2	8	10	10	13	8	8	8	8	11	10	4	7	6	14	12
U Experts	9.6	4	10	14	1	9	3	11	8	11	12	14	4	14	14	3	13	14	14	14
U Generalists	10.9	5	12	15	1	13	14	13	13	9	7	15	5	12	15	3	14	11	15	15
U Gen+Experts	9.3	5	8	14	1	6	4	13	8	8	11	14	4	14	14	3	9	14	14	14
U R18/34/50 Exp.	9.1	3	10	14	1	6	3	11	8	7	11	14	4	11	14	2	13	14	14	14
U R101 Experts	9.7	6	9	14	1	10	5	10	8	8	11	14	6	13	14	2	11	13	14	13
C HyperExperts	9.9	5	13	15	1	4	7	13	12	9	12	15	2	14	15	2	8	11	15	15
C AugExperts	10.3	4	8	15	1	9	7	15	9	9	8	15	5	15	15	4	11	15	15	15

enough to yield performant models. Figure 6 shows this is not the case. All categories perform worse, though it is particularly notable in Natural datasets where the ensembles lose the edge of relevant pre-training.

Following from the results of Puigcerver et al. (2020), it is clear that the choice of expert is important in the low-data regime. The final results are much more significantly influenced by the initialisation, and transfer from pre-training, than when more data is available. Less suited pre-trained models will lead to lower accuracy and ultimately drag down the predictive power of the entire ensemble, hence why when suitability of pre-trained models varies significantly, the k NN becomes much more valuable.

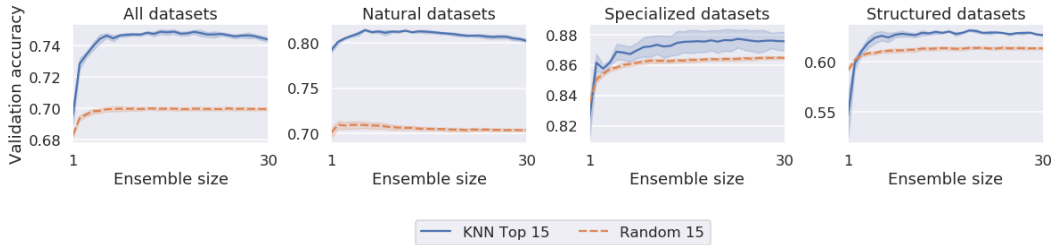


Figure 6: Comparison of ensembles created from diversity generated via a random selection of pre-trained models vs using k NN to filter pre-trained models. Diverse initialisations alone are not enough; we require individually competitive models.

C.4 EVALUATING FILTERING WITH k NN

To evaluate the performance of using k NN as a strategy to narrow down a set of pre-trained models, we computed the ensemble test performance on VTAB_{1K} using the 244 JFT R50 pre-trained experts with the following strategies:

Brute + Greedy Ensemble: finetune each of the 244 available pre-trained models using the default sweep (4 hyper-parameters). The 976 obtained models are then used to build an ensemble of 15 models using the greedy algorithm.

k NN top 15 + Greedy Ensemble: use k NN to select 15 out of the 244 available pre-trained models. The 15 models are finetuned 4 times obtaining 60 models that are then used to build an ensemble of 15 models using the greedy algorithm.

k NN + Threshold Ensemble: use k NN to select a number (≤ 15) of pre-trained models per task using the threshold approach described in Section 2.3.1. Each selected models is finetuned 4 times with the default sweep and the one with the best validation accuracy is used in the final ensemble.

The results in Table 5 show that k NN filtering was effective at reducing the number of models without drop in VTAB_{1K} accuracy. Note also that k NN based approaches were able to reduce the noise from the 244 models so that the final ensemble (computed over the validation score) obtained better test accuracy for the natural datasets where we expect to exist clear experts among the models. This conclusion is made stronger by the fact that the k NN + Threshold Ensemble was also able to beat the brute approach.

Table 5: VTAB_{1K} test accuracy of different selection and ensemble strategies. Note that *Brute + Greedy Ensemble* requires finetuning 976 models per task whether the others require 60 or less.

	VTAB _{1K}	Natural	Specialised	Structured
Brute + Greedy Ensemble	76.20	81.17	85.46	67.22
k NN top 15 + Greedy Ensemble	76.41	82.20	85.55	66.79
k NN + Threshold Ensemble	76.45	82.00	85.90	66.87

C.5 FILTERING UPSTREAM DIVERSITY

As discussed in Section 2.3.1 by keeping only the pre-trained models with top- $\tau\%$ k NN accuracy instead of all the top-K it is possible to free finetune budget that can be used to generate downstream diversity. This threshold mechanism when applied to the JFT-R50 pre-trained models only had effect in 7 of the 19 datasets, 6 of them in natural (where we expected to have experts pre-trained on related tasks which are a really good fit); in the remaining ones, the maximum cap was applied. Figure 7 shows the improvements on those datasets. For structured and specialised datasets where pre-training data is generally much less related to the downstream task, k NN accuracies are more uniform as reported in Puigcerver et al. (2020). Thresholding there tends to just keep all the experts.

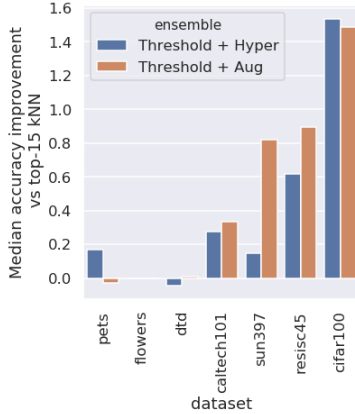


Figure 7: Median test accuracy improvement from using the threshold mechanism on JFT-R50 pre-trained models. We show only datasets where the thresholding mechanism had an effect.

C.6 GREEDY PRE-TRAINED MODEL SELECTION

We experimented with a variant of the k NN selection which aimed to pick pre-trained models that would ensemble well together (Greedy), as opposed to picking pre-trained models which are independently accurate (top-K). The effect of this for JFT and ImageNet21k ResNet-50 experts is shown in Figure 8. The greedy approach does not give any clear benefits, and we found at test time it resulted in a small drop in performance. We suspect this is due to the small amount of data the k NN is making decisions based off.

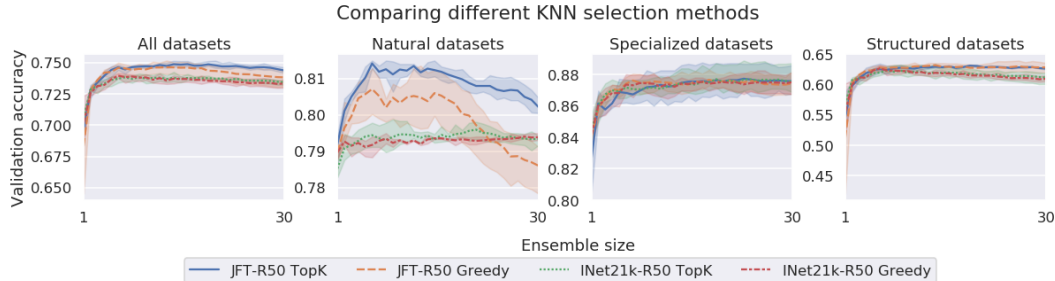


Figure 8: Comparison of greedy k NN and top-K k NN selection.

C.7 SELECTING FROM MULTIPLE SCALES OF PRE-TRAINED MODELS

Figure 9 shows validation accuracy performance of ensembles of different architecture sizes. Firstly, we note again that to some extent, performance gains stack with architecture size. Secondly we note that the larger architectures are not always better; for instance, the ResNet-34s perform significantly better than others on structured datasets. Lastly, when considering the ‘All’ line — where the k NN selected from experts pretrained at all four architecture scales — we would hope it performs as well as the best alternative line. To some extent it does, but it does not always get it right.

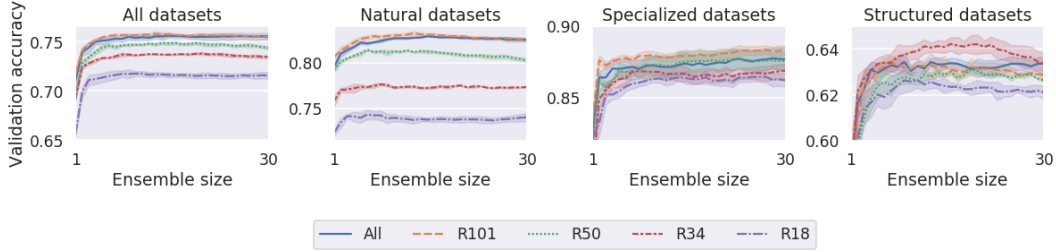


Figure 9: Expert ensembles created from pretraining on JFT subsets at architecture scales. ‘All’ indicates an ExpertEnsemble where the k NN selected from all architecture sizes.

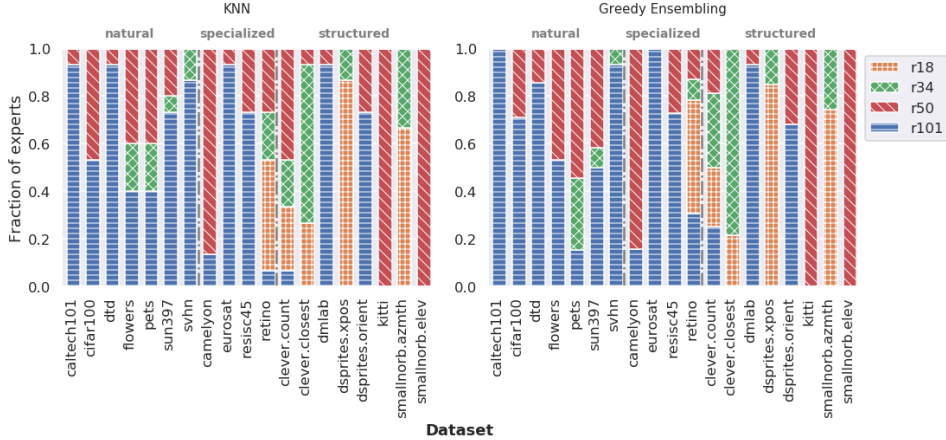


Figure 10: Experts selected when picking from JFT ResNet experts with 18, 34, 50 and 101 layers

One may expect that the typically higher performing ResNet-101s (Kolesnikov et al., 2019) would be heavily favoured by the k NN algorithm; or failing that, after finetuning on the dataset, would be more performant on the validation set and therefore favoured by the greedy algorithm. Interestingly - on both fronts - this is not the case. The approach is often creating ensembles of diverse architectures scales. We discuss further in Appendix C.8, but it is not clear there is always a clear gain from such diversity. In fact, we see here the k NN is not picking as many R34 experts as it should be, despite their clear superiority after fine-tuning; this may be due to comparing features with different dimensions (ResNet-18s and 34s produce 512 dimension features, whereas ResNet-50 and 101 produce 2048 dimensional features), and we suspect using a different distance metric (e.g. cosine distance instead of Euclidean) may have alleviated this.

C.8 SUBSET SELECTION FROM A VERY LARGE POOL

For the majority of our experiments, the k NN picked $K = 15$ feature extractors from a pool of 244 (for JFT experts) or 50 (for ImageNet21k experts). Intuitively, the larger the pool of candidate feature extractors is, the harder the job for the k NN; and it is clear from the marginal drop in performance when combining the two pools of feature extractors that the task of subset selection will become harder as more candidates are added.

We consider what happens in the extreme limit - as discussed in Appendix A.1, as well as the 50 ImageNet21k experts, we trained 4 different architectures of ResNet on 244 different slices of JFT with 2 different pretraining methods (fine-tuning from a generalist and training from scratch). This gives us a total of 2002 experts in our pool. Figure 11 shows the experts picked from this pool. Notably, once again, both the k NN and the greedy algorithm pick a diverse selection of models. There are some peculiarities; e.g. for some datasets, ImageNet21k-trained experts perform better as a k NN classifier, whereas the JFT experts are better after finetuning, raising issues when using the former as a proxy for the latter.

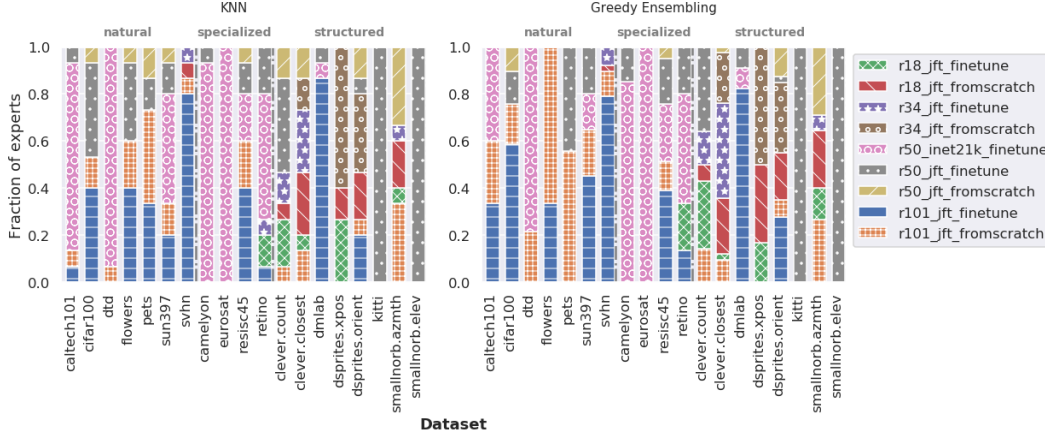


Figure 11: Experts selected when picking from all available feature extractors.

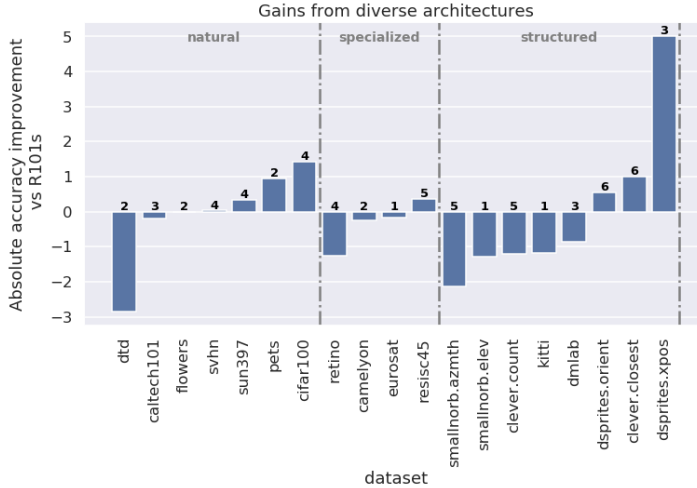


Figure 12: Test performance of experts selected from all feature extractors vs. those only selected from R101s. Numbers show the number of unique ‘types’ of experts picked; e.g. for dtd, the final ensemble contains ResNet-50s trained on ImageNet21k and ResNet-101s trained on JFT. There is no clear correlation between the number of unique expert types (a form of diversity) and the accuracy.

The interesting question is: are we benefiting from diverse architectures? We suspect a-priori that the best-performing models will be the R101 experts trained on JFT. We therefore compare the performance of the diverse expert ensembles with the R101 ones - this is shown in Figure 12. It is not immediately clear the existence of any correlation between the diversity of ensembles – in terms of architecture etc. – with the final test performance. That is not to say there is no theoretical benefit to that at all; this is obfuscated by the extra k NN stage, which means there are almost certainly more optimal combinations of different architectures/pretraining styles.

C.9 AVOIDING OVERFITTING

All ensembling approaches attempted in this paper experienced higher drops in performance from validation to the test set. It has been previously noted that, especially with small validation datasets, the ensembling process itself can overfit to the validation data (Caruana et al., 2004). We attempted a few methods to alleviate this:

Bootstrapping data: At each iteration of the greedy algorithm, we randomly bootstrap the 200 validation data points, with some proportion M (such that $200M$ samples are used to make ensembling decisions). This should ensure the ensembling process is not overfitting to individual data points too much, a phenomenon we observed when constructing greedy ensembles based on accuracy. This helped marginally when selecting ensembles for maximum accuracy, but we did not see any improvement when using this to select ensembles for maximum cross-entropy.

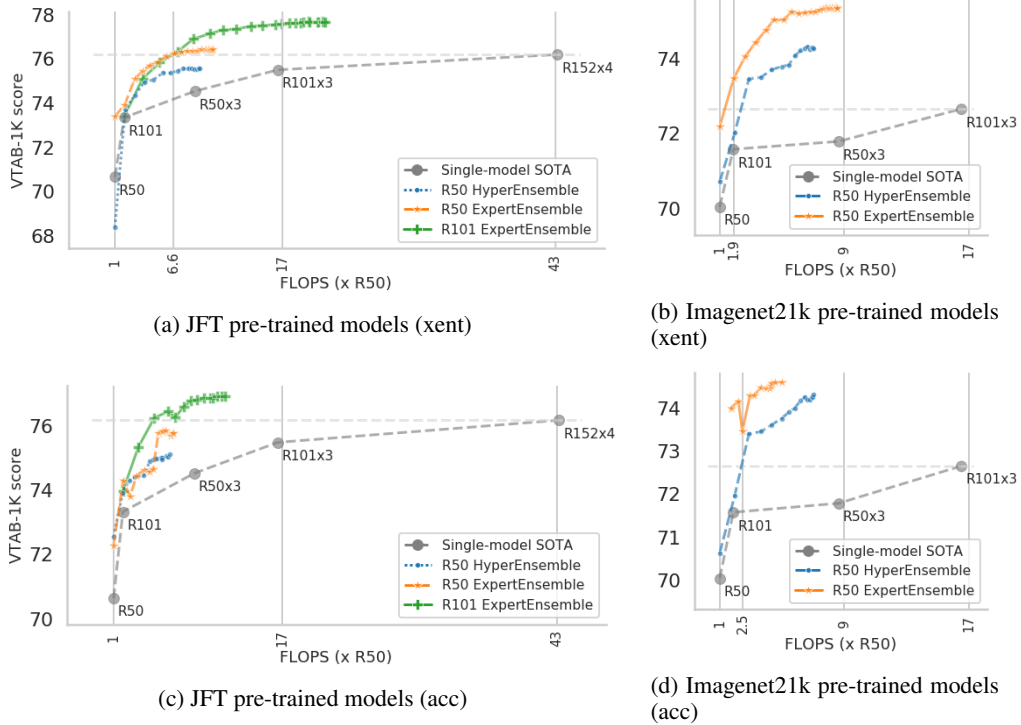


Figure 13: Effect of different greedy ensembling selection strategies. We vary the inference budget to generate ensemble curves. Note that when selecting based on accuracy, the ensembles perform better with smaller ensembles, but that performance doesn’t monotonically increase with increasing budget, and overall performance is worse.

Greedy selection with replacement: Caruana et al. (2004) reported that allowing the algorithm to re-select models it had already selected reduced overfitting. The extra benefit (in our view, which is not discussed in that work) is that by weighting ensembles according to the number of times they were picked, we can use a weighted average, downweighting less performant models and potentially improving performance. In our set up unfortunately, neither of these held; it did not reduce overfitting, and weighting did not improve performance.

Ensembling to maximise cross-entropy: We initially constructed ensembles in order to maximise accuracy on the validation set. This resulted in multiple problems; with so few data points, ensembles were making decisions based on single data points (fighting for 0.5% improvement in accuracies on the 200 data point validation set). Furthermore, multiple ensembles generated by the greedy process would give the exact same validation accuracy, resulting in the need for an arbitrary tiebreaker. For all model sets, with upstream or downstream diversity, maximising cross entropy generalised better and avoided the latter problem, and was the most successful approach to avoiding over-fitting. As well as overall improving performance, we found it more reliable - when making ensembling decisions based on cross-entropy, final test performance monotonically increased with the inference budget (ensemble size). It is worth noting however that decisions made based on cross-entropy are worse for single models or smaller ensembles. With a small inference budget, we suggest optimising for accuracy. These phenomena are evident in Figure 13.

Note that Caruana et al. (2004) suggest bagging of *ensembles* - i.e. selecting multiple ensembles using different splits of the data, then combining them in one pool. We did not opt for this, as we wanted to have a fixed downstream inference budget, and it is non-trivial to ensure that when combining multiple ensembles of variable size.