A Appendix

A.1 Detailed Results on Each Dataset

Biased MNIST. In Table A6, we present the unbiased test accuracies and majority/minority group accuracies for each bias variable. Methods run on Occam ResNet-18 lower the majority/minority discrepancy (MMD) compared to the methods run on ResNet-18 for all of the variables, indicating that OccamNets lower the tendencies to latch onto all of the spurious factors. OccamResNet-18 is especially robust to letter, letter color and texture color biases as shown by the low MMD values of $0 - 2.8\%$ compared to the larger MMD values obtained by ResNet-18.

Table A6: Accuracies on majority (maj)/minority (min) groups for each bias variable in Biased MNIST ($p_{bias} = 0.95$).

Architecture+Method		Digit Scale	Digit Color	Texture	Texture Color	Letter	Letter Color
	Test Acc.	maj/min	maj/min	maj/min	maj/min	maj/min	maj/min
Results on ResNet-18							
ResNet+ERM	36.8	87.2/31.3	78.5/32.1	76.1/32.4	41.9/36.3	46.7/35.7	45.7/35.9
ResNet+SD [51]	37.1	83.4/32.0	76.9/32.7	76.7/32.7	42.3/36.6	48.3/35.8	48.9/35.9
ResNet+UpWt	37.7	88.0/32.1	80.4/32.9	75.6/33.4	41.9/37.2	46.7/36.6	46.9/36.7
ResNet+gDRO [56]	19.2	55.0/15.2	50.2/15.7	63.4/14.2	24.8/18.6	26.7/18.3	29.5/18.1
ResNet+PGI [3]	48.6	91.9/43.8	84.8/44.6	79.5/45.1	51.3/48.3	67.2/46.5	55.8/47.9
Results on OccamResNet-18							
OccamResNet	65.0	94.6/61.7	96.3/61.5	81.6/63.1	66.8/64.8	64.7/65.1	64.7/65.1
OccamResNet+SD [51]	55.2	92.3/51.1	92.9/50.9	78.9/52.5	57.4/54.9	55.9/55.1	55.3/55.2
OccamResNet+UpWt	65.7	95.1/62.5	96.3/62.3	82.4/63.9	68.3/65.5	65.3/65.8	65.3/65.8
OccamResNet+gDRO [56]	29.8	72.8/25.0	69.5/25.3	45.8/28.0	39.4/28.8	29.7/29.8	36.1/29.1
OccamResNet+PGI [3]	69.6	95.4/66.7	97.0/66.5	88.6/67.4	71.4/69.4	69.6/69.6	70.5/69.5

COCO-on-Places. In Table A7, we present the accuracies on each of the test splits of COCO-on-Places, alongside the average precision for the anomaly detection task. As discussed in 5.1, methods run on OccamResNet-18 show improvements over the methods run on ResNet-18 on the shifted test splits and the anomaly detection task. Furthermore, while PGI run on ResNet-18 shows a large drop of 7.4% on the in-distribution test split, methods (barring gDRO) run on OccamResNet-18 show smaller drops of $0.1 - 2.1\%$, indicating robustness to distributions consisting of the same or different biases as compared to the train distribution.

BAR. First of all, BAR consists of only 1941 samples, so we pre-trained ResNet-18 and OccamResNet-18 on 100 classes of ImageNet (obtaining 92.6% and 92.1% top-5 accuracies respectively) before training on BAR. Without the pre-trained weights, BAR obtains 15-20% lower test set accuracies for both ResNet and OccamResNet as compared to the results with pre-trained weights.

Table A7: Accuracy on the three splits of COCO-on-Places, alongside average precision for the anomaly detection task.

Methods	In-Distribution	Unseen Backgrounds	Seen, but Non-Spurious Backgrounds	Anomaly Detection
Results on ResNet-18				
ResNet+ERM	**84.9** ± 0.5	53.2 ±0.7	35.6 ±1.0	20.1 ±1.5
ResNet+SD [51]	**85.3** ±0.3	52.8 ±0.9	35.4 ±0.5	19.9 ±1.4
ResNet+UpWt	**84.9** ±0.6	52.3 ± 0.7	35.2 ± 0.4	20.4 ±1.9
ResNet+gDRO [56]	81.6 ±0.7	49.3 ±1.3	35.3 ±0.1	19.6 ±1.7
ResNet+PGI [3]	77.5 ±0.6	52.8 ±0.7	42.7 ±0.6	20.6 ±2.1
Results on OccamResNet-18				
OccamResNet	84.0 ±1.0	**55.8** ±1.2	**43.4** ±1.0	**22.3** ±2.8
OccamResNet+SD [51]	84.8 ±0.4	55.3 ±0.5	39.4 ±0.6	20.3 ±1.0
OccamResNet+Up Wt	82.9 ±0.5	**56.6** ±1.0	42.9 ±0.8	21.0 ±0.9
OccamResNet+gDRO [56]	78.6 ±0.7	50.7 ±2.0	40.7 ± 1.5	19.3 ±2.3
OccamResNet+PGI [3]	82.8 ±0.6	55.3 ±1.3	**43.6** ± 0.6	**21.6** ±1.6

Now, as shown in Table A8, methods run on OccamResNet show gains in terms of the overall test set accuracies over the methods run on ResNet. The per-class standard deviations are larger (1.8-16.2%) as compared to the standard deviations for the overall test set accuracies (0.7-2.4%). That is, across the five different experiments run with different random seeds, the same methods run on the same architectures end up favoring different classes. We hypothesize that despite starting off from the same initial conditions i.e., the same pre-trained parameters, the randomness in the mini-batches drive the models to favor certain classes over the others. Tuning the optimizer e.g., switching to SGD, lowering the learning rates or increasing the weight decay can potentially help mitigate the unstable behavior.

A.2 Early Exit Statistics

To examine the efficiency and robustness of each exit for all of the datasets, we present the exit %, accuracy on the exited samples and overall exit-wise accuracies on all the samples for OccamResNet-18 in Table A9. For Biased MNIST, the earliest exits E_1 and E_2 have high exit percentages of 59.8% and 26.9% respectively, alongside high accuracies on the exited samples: 68.1% and 64.8% respectively. These results show that OccamResNet has learned to identify and trigger earlier exits whenever appropriate. For COCO-on-Places, we observe large accuracies of 50.8% and 50.2% on the 13.9% and 49.6% samples exited from E_2 and E_3 respectively. The large percentage of samples exiting from E_3 shows that OccamResNet is capable of using the full network depth whenever needed. The accuracy on the samples that exited from E_1 is however low: 31.3%, even though

Table A8: Overall and per-class accuracies on BAR

Methods	Overall	Climbing	Diving	Fishing	Pole Vaulting	Racing	Throwing
Results on ResNet-18							
ResNet+ERM	51.3 ±1.9	**69.5** ±7.5	29.2 ±1.8	39.9 ±16.2	55.5 ±6.4	**75.6** ±5.6	**31.8** ±4.3
ResNet+SD [51]	51.3 ±2.3	62.1 ±7.5	35.8 ±2.0	51.2 ±6.4	62.4 ±9.2	71.6 ±10.0	18.5 ±6.7
ResNet+Up Wt	51.1 ±1.9	61.7 ±13.2	**43.9** ±5.8	42.3 ±8.3	52.3 ±7.4	67.9 ±6.7	**28.2** ±12.8
ResNet+gDRO [56]	38.7 ±2.2	49.5 ±8.5	40.3 ±8.4	44.0 ±10.4	39.9 ±7.1	41.7 ±4.0	13.5 ±5.9
ResNet+PGI [3]	**53.6** ±0.9	61.2 ±10.4	38.4 ±4.1	42.9 ±8.4	**73.3** ±3.7	68.9 ±5.9	23.5 ±1.9
Results on OccamResNet-18							
OccamResNet	52.6 ±1.9	59.3 ±3.8	42.3 ±7.5	44.6 ±14.9	60.5 ±8.6	74.1 ±7.2	22.1 ±3.9
OccamResNet+SD [51]	52.3 ±2.4	56.4 ±6.8	34.3 ±5.8	**55.4** ±7.4	**69.1** ±4.9	72.9 ±4.2	21.8 ±2.1
OccamResNet+Up Wt	52.2 ±1.4	57.9 ±1.8	35.7 ±7.5	51.8 ±11.2	64.3 ±8.8	71.8 ±3.8	27.4 ±3.5
OccamResNet+gDRO [56]	52.9 ±0.8	51.2 ±9.6	42.8 ±8.2	52.3 ±5.1	63.5 ±7.3	**74.2** ±5.2	25.3 ±4.5
OccamResNet+PGI [3]	**55.9** ±0.7	**64.2** ±5.1	**52.3** ±6.4	51.4 ±8.3	64.4 ±4.1	70.9 ±8.1	18.6 ±6.8

the overall accuracy is 42.4%, indicating need for improvement in terms of training the earlier exit gates. We believe that tuning the training thresholds more comprehensively can potentially close this gap. Finally, for BAR, more than half i.e., 55.9% of the samples exit from E_3. The accuracies on the samples exited from E_1 and E_2: 55.0% and 65.3% are higher than the overall accuracies computed on all the samples i.e., 47.4% and 52.3% respectively. This again shows the ability to exit early whenever appropriate and the ability to utilize the full network depth only for the remaining samples.

Table A9: Percentage samples exited: (Exit %), accuracy (Acc.) on exited samples and accuracy on all the samples for each exit (E_j).

	Biased MNIST				COCO-on-Places				BAR			
	E_0	E_1	E_2	E_3	E_0	E_1	E_2	E_3	E_0	E_1	E_2	E_3
Exit%	0.0	59.8	26.9	13.3	0.0	36.5	13.9	49.6	0.0	23.7	20.3	55.9
Acc. (exited)	N/A	68.1	64.8	52.1	N/A	31.3	50.8	50.2	N/A	55.0	65.3	46.6
Acc. (all)	12.7	65.1	65.5	65.5	10.0	42.4	43.4	41.4	26.5	47.4	52.3	52.5

Table A10: Exit comparison on models trained with different levels of biases on Biased MNIST. We present the percentage samples exited: (Exit %) and accuracy on all the samples for each exit (E_j).

	$p_{bias} = 0.5$				$p_{bias} = 0.95$			
	E_0	E_1	E_2	E_3	E_0	E_1	E_2	E_3
Exit%	0.0	67.3	23.8	8.9	0.0	53.6	34.2	12.2
Acc. (all)	N/A	98.2	98.1	98.1	N/A	63.3	63.3	62.6

Table A11: Exit comparison on different test splits of COCO-on-Places. We present the percentage samples exited: (Exit %), accuracy (Acc.) on exited samples and accuracy on all the samples for each exit (E_j).

	In-Distribution				Unseen Backgrounds				Unbiased Backgrounds			
	E_0	E_1	E_2	E_3	E_0	E_1	E_2	E_3	E_0	E_1	E_2	E_3
Exit%	0.0	54.6	12.4	33.0	0.0	13.1	9.7	77.2	0.0	39.4	14.5	46.1
Acc. (exited)	N/A	94.7	86.5	66.9	N/A	58.9	67.8	52.3	N/A	27.5	55.4	49.2
Acc. (all)	67.2	81.1	84.0	85.3	20.9	51.7	55.1	55.4	11.2	40.1	41.6	39.4

Exit Statistics on differently shifted distributions.

In general, we find that earlier exits are triggered more often for in-distribution (easier) test samples as compared to shifted distribution (more difficult) test samples. As shown in Table A10, for BiasedMNIST, when p_{bias} is increased from 0.5 (easy) to 0.95 (hard), exit% of the earliest exit: E_1 drops from 67.3% to 53.6%. Similarly, as shown in Table A11, for COCO-on-Places, E_1's exit% on in-distribution (easy) split is 54.6%, whereas it is 39.4% on unbiased backgrounds (hard) split. However, for the test split with unseen backgrounds, E_1's exit%: 13.1% is lower than the exit% of 39.4% obtained for the test split with unbiased backgrounds, despite the latter being more difficult. OccamNet failed to trigger earlier exits even though E_1 was accurate for 51.7% of the samples E_2 was comparable with E_3 in terms of overall accuracy. We hypothesize that the earlier exits failed to trigger since they had not been trained to exit when the backgrounds are out-of-distribution. Thus, one area for improvement is to enable the ability to exit confidently in spite of the presence of previously unseen factors. Note that this analysis was performed on a single run, so may have small differences with the multi-run averages presented elsewhere.

A.3 Using comparable # of parameters

In the main paper, we compared OccamResNet-18 with 8M parameters (feature width = 48) and ResNet-18 with 12M parameters (feature width = 64). To examine if the lower number number of parameters is helping e.g., due to implicit regularization, we test an OccamResNet-18 with 12M parameters by setting the feature width to 58. As shown in Table A12, OccamResNet-18 with 12M parameters shows small improvements over OccamResNet-18 with 8M parameters in all the datasets. A more thorough analysis of model sizes and their impacts on accuracy is an interesting study and we leave this to future work.

A.4 Sample Complexity

It is desirable to have models that generalize despite being trained with a limited number of samples i.e., with reduced sample complexity. This is especially true for biased datasets, where reducing the train set size can amplify biases [72]. To study the ability to generalize when only a subset of the training data is available,

Table A12: We train on OccamResNet-18-width-58 with 12M parameters (feature width set to 58) to make the number of parameters comparable to ResNet-18 (12M parameters, feature width=48).

Architecture+Method	Biased MNIST	COCO-on-Places	BAR
Results on Standard ResNet-18 (12M parameters, feature width=64)			
ResNet+ERM	36.8 ± 0.7	35.6 ± 1.0	51.3 ±1.9
ResNet+SD [51]	37.1 ± 1.0	35.4 ± 0.5	51.3 ±2.3
ResNet+Up Wt	37.7 ± 1.6	35.2 ± 0.4	51.1 ±1.9
ResNet+gDRO [56]	19.2 ± 0.9	35.3 ±0.1	38.7 ±2.2
ResNet+PGI [3]	**48.6** ± 0.7	**42.7** ± 0.6	**53.6** ±0.9
Results on OccamResNet-18 (8M parameters, feature width=48)			
OccamResNet	**65.0** ±1.0	**43.4** ± 1.0	52.6 ±1.9
Results on OccamResNet-18-width-58 (12M parameters, feature width=58)			
OccamResNet	**65.9** ±1.3	**43.8** ± 1.1	**53.5** ±2.2

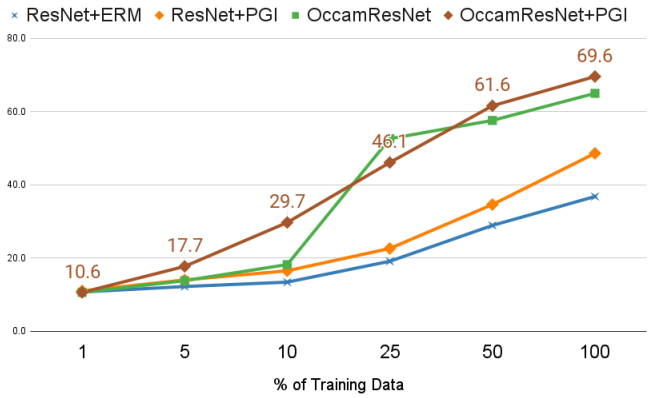

Fig. A6: Unbiased accuracies obtained when trained with the indicated percent of training data.

we train ResNet and OccamResNet on $1\%, 5\%, 10\%, 25\%, 50\%, 100\%$ of Biased MNIST's train set. As shown in Fig. A6, OccamResNet (without PGI) trained on only 25% of the data outperforms ResNet+ERM and ResNet+PGI trained on 100% of the data showing increased sample complexity. When trained on only 10% of the training set, OccamResNet+PGI outperforms rest of the methods by large margins of $11.5 - 16.2\%$ showing that OccamResNet with group labels show the greatest efficacy in the low-shot data regime. When only 1% of the training data is available, all the methods obtain chance-level accuracies (i.e., near 10%) indicating lack of enough sufficient training samples for classification. For the rest, methods run on OccamResNet-18 outperform the methods run on ResNet-18, showing improved sample complexity.

A.5 Robustness to Varying Levels of Bias in Biased MNIST

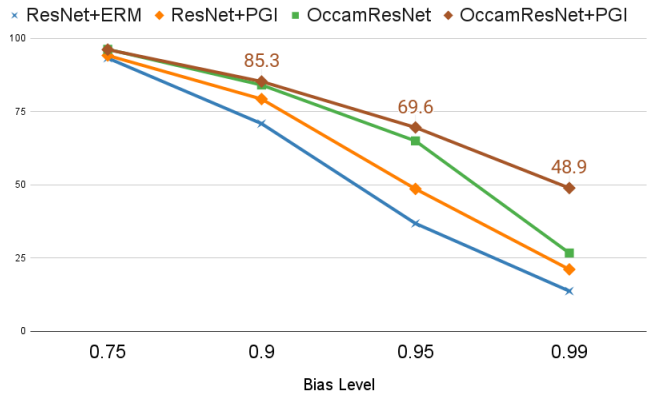

Fig. A7: Unbiased accuracies at varying bias levels (p_{bias}) in Biased MNIST.

To gauge the robustness of models, it is important to examine their behaviors across varying levels of biases. For this, we present the unbiased accuracies obtained by training separate models on training sets with $p_{bias} \in \{0.75, 0.9, 0.95, 0.99\}$. As shown in fig. A7, all of the methods obtain similar accuracies at $p_{bias} = 0.75$, where bias is not severe. OccamResNet+PGI outperform rest of the methods at $p_{bias} = \{0.9, 0.95, 0.99\}$. The gap between OccamResNet+PGI and other methods are especially drastic for $p_{bias} = 0.99$, indicating that when OccamResNet is trained to have similar prediction distributions across groups, it is capable of tackling highly biased training distributions too.

A.6 Evaluation on Other Architectures

Apart from ResNet, we also tested the proposed inductive biases on EfficientNet and MobileNet. The results are presented in Table A13. For both Biased MNIST and COCO-on-Places, Occam variants outperform the standard architectures, showing the efficacy of the proposed modifications.

A.7 OccamNet Implementation Details

In OccamNet, each exit module E_j takes in feature maps produced by the corresponding block B_j of the backbone network. E_j consists of two 3×3 convolutional layers (F_j) for the initial pre-processing of the feature maps. F_j consists of convolutional layers with the number of channels set to: $max(\frac{d_j}{4}, d_{min})$, where d_j is the number of channels in the feature maps produced by B_j, and d_{min} is set to 32 for OccamResNet and OccamMobileNet and 16 for OccamEfficientNet. Feature

Table A13: Unbiased test set accuracies comparing the standard and Occam variants of ResNet-18, EfficientNet-B2 and MobileNetv3 architectures, run without additional debiasing procedures.

Architecture	Number of Parameters	Biased MNIST	COCO-on-Places
ResNet-18	12M	36.8	35.6
OccamResNet-18	12M	65.9	43.8
EfficientNet-B2	9M	34.4	34.2
OccamEfficientNet-B2	9M	59.2	39.2
MobileNet-v3	5.5M	40.4	34.9
OccamMobileNet-v3	5.5M	49.9	40.1

maps from F_j are fed into the CAM predictor C_j and the exit gate G_j. C_j is a 1×1 convolutional layer with the number of output channels set to the number of classes, n_Y. G_j consists of a 16-dimensional hidden ReLU layer followed by a sigmoid layer that predicts the exit probability.

Exit Details. For convenience, we specify the exit locations with reference to PyTorch 1.7.1 implementations of the architectures. For ResNet, the residual layers that yield the same number of output channels are grouped together and we refer to each of those groups as a 'block'. ResNet-18 consists of 4 blocks and we attach an exit to each of the blocks. For OccamResNet-18, with feature width of 58, the exit-wise input dimensions are: $E_0 : 58$, $E_1 : 116$, $E_2 : 232$ and $E_3 : 464$. Similarly, EfficientNet-B2 consists of 9 blocks and we attach the exits to the $3^{rd}, 5^{th}, 7^{th}$ *and* 9^{th} blocks. We decrease the width multiplier of 1.1 in the standard architecture to 0.88 in OccamEfficientNet-B2 to create a model with comparable number of parameters of 9M for both. The input dimensions of the corresponding exits are: $E_0 : 24$, $E_1 : 72$, $E_2 : 168$ and $E_3 : 1120$. Finally, MobileNetv3-large consists of 17 blocks, and the exits are attached to the $2^{nd}, 7^{th}, 13^{th}$ and the 17^{th} blocks. We decrease the width multiplier from a value of 1 in MobileNet-v3-large to 0.95 in OccamMobileNet-v3-large, so that both models have 5.5M parameters. The input dimensions of the corresponding exits are: $E_0 : 16, E_1 : 40, E_2 : 104, E_3 : 912$.

Modifications for COCO-on-Places. For COCO-on-Places, the images are small (64×64), so for ResNet-18 and OccamResNet-18, we replace the first convolutional layer (kernel size=7, padding=3, stride=2), with a smaller layer (kernel size=3, padding=1 and stride=1) and also remove the initial max pooling layer. For the standard and Occam variants of EfficientNet-B2 and MobileNet-v3, we scale up the image size to 224×224, which improved the accuracy.

Computational Costs and Training Durations OccamResNet18 incurs additional multiply-accumulate (Mac) operations, requiring 2.11 GMacs compared to 1.82 GMacs required by ResNet18. While OccamResNet is slower than ERM on ResNet, it is faster than PGI run on ResNet. Average training durations per epoch for ERM on ResNet, PGI on ResNet and OccamResNet are: 10,

17, 14 secs for COCO-on-Places and 40, 67, 60 secs for BiasedMNIST on a Titan RTX.

A.8 Hyperparameters and Other Settings

In Table A14, we present the details about optimizers, training epochs and other hyperparameters for each method on each dataset. The hyperparameter search grids for OccamResNet-18 and all of the comparison methods are shown below. For each dataset, we tune ResNet-18 and OccamResNet-18 separately.

Spectral Decoupling (SD). The output decay term λ is used to penalize the model predictions by using a regularizer: $\frac{\lambda}{2}||\hat{y}||_2^2$. We search for the output decay term $\lambda \in \{1, 0.1, 0.01, 10^{-3}, 10^{-4}\}$.

Group Upweighting (UpWt). Group adjustment hyperparameter, i.e., the exponentiation factor γ is used to balance the group-wise contributions $\frac{1}{n_g^\gamma}$, where n_g is the number of samples in group g. We search for $\gamma \in \{0.5, 1, 2, 3\}$.

Group DRO (gDRO). Again, we search for the group adjustment hyperparameter, i.e., $\gamma \in \{0.5, 1, 2, 3\}$. Group weight step size, which is used to control the group-wise loss weights is selected by searching from these values: $\{0.1, 0.01, 10^{-3}, 10^{-4}\}$.

Predictive Group Invariance (PGI). The search range for the invariance penalty loss, i.e., the KLD loss between different groups from the same class is: {1, 10, 50, 100, 500, 1000}.

OccamNets. For OccamNets, we recommend tuning the accuracy threshold of the first exit: ($\tau_{acc,0}$) on a validation set, but fixing rest of the hyperparameters, based on the following observations:

- *Bias Amplification Factor (γ_0) and Weight offset (ϵ):* We tuned γ_0 and ϵ on COCO-on-Places, but fixed the values for rest of the datasets. We observe that $\gamma_0 \geq 3$ ensures sufficient bias amplification, so recommend using $\gamma_0 = 3$. Furthermore, $\epsilon = 0.1$ ensures non-zero losses in all the datasets, so we recommend using this default value.
- *Accuracy Thresholds (τ_{acc}):* We use arithmetic progression for the mean-per-class accuracy thresholds τ_{acc}, with the difference $\Delta\tau_{acc,j}$, set to 0.1, i.e., the threshold is increased by 0.1 every subsequent exit. We search for the initial training threshold $\tau_{acc,0} \in \{0.1, 0.3, 0.5\}$. BMNIST and COCO were relatively insensitive to $\tau_{acc,0}$, with absolute differences of only: 1-2% in accuracy and 1-4% in exit%. For BAR and ImageNet, we decreased $\tau_{acc,0}$ to 0.1 since higher values increase exit% of E_1, which decreases the overall accuracy. So, we recommend tuning $\tau_{acc,0}$.
- *Normalization term:* The balancing/normalization in equation 4 can be generalized as $(\frac{1}{\sum_{1(g_j=k)}})^\beta$. We searched for β in $\{0.5, 1.0\}$. With $\beta = 1$, only 8.8% samples exited from E_1 for BMNIST, compared to 59.8% with $\beta = 0.5$. Accuracies for β =0.5/1.0 were similar: e.g., 65.0%/64.2% for BMNIST and 44.1%/44.0% for COCO (single runs), so we chose $\beta = 0.5$ to favor earlier exits on all the datasets.

Table A14: Hyperparameters and other settings used for each method on all of the datasets.

Datasets/ Methods	**Setting**	**Biased MNIST**	**COCO on Places**	**BAR**
Common to all the methods	Optimizer	Adam	SGD	Adam
	Learning Rate (LR)	10^{-3}	0.1	5e-4
	LR Decay Milestones	[50,70]	[100,120,140]	-
	LR Decay Gamma	0.1	0.1	-
	Weight Decay	5×10^{-4}	5×10^{-4}	5×10^{-4}
	Momentum	-	0.9	-
	Batch Size	128	64	128
	Epochs	90	150	150
Spectral Decoupling (SD) on ResNet-18 [51]	Output Decay (λ)	$\lambda = 0.1$	$\lambda_{min.} = 10^{-3}$ $\lambda_{maj.} = 0.1$	$\lambda = 0.1$
Spectral Decoupling (SD) on OccamResNet-18 [51]	Output Decay (λ)	$\lambda = 10^{-3}$	$\lambda_{min.} = 10^{-3}$ $\lambda_{maj.} = 0.1$	$\lambda = 10^{-3}$
Up Wt	Exponentiation Factor (γ)	2	1	1
Group DRO (gDRO) on ResNet-18 [56]	Step size	10^{-3}	10^{-3}	0.01
	Exponentiation Factor (γ)	0.5	1	0.01
Group DRO (gDRO) on OccamResNet-18 [56]	Step size	10^{-3}	10^{-4}	0.01
	Exponentiation Factor (γ)	0.5	0.5	0.5
Predictive Group Invariance (PGI) on ResNet-18 [3]	Invariance Loss Weight	100	50	10
Predictive Group Invariance (PGI) on OccamResNet-18 [3]	Invariance Loss Weight	50	1	50
OccamNets	Threshold for E_0 ($\tau_{acc,0}$)	0.5	0.5	0.1
	CAM Suppression Loss Weight (λ_{CS})	0.1	0.1	0.1

A.9 Issues Training with GroupDRO (gDRO)

We find that gDRO on Biased MNIST and ResNet+gDRO on BAR obtain accuracies lower than ResNet+ERM. To alleviate this issue, we tried to tune the hyperparameters by lowering the learning rates to $\{10^{-4}, 10^{-5}\}$ and increasing the weight decays to $\{0.1, 0.01, 10^{-3}\}$ as suggested in [56], yet gDRO obtained low accuracies. We believe the challenge stems from the large number of dataset groups in Biased MNIST and the small training set size of BAR. While optimizing gDRO on such conditions still remains a challenge, gDRO run on Occam-

ResNet showed accuracy gains of 10.6% on Biased MNIST and 14.2% on BAR over gDRO run on ResNet, indicating that OccamNets also offer better training process.

A.10 Augmentations

Biased MNIST. We do not perform any augmentation.

COCO-on-Places. Following [3], we apply random cropping by padding the original images by 8 pixels on all the sides (reflection padding) and taking 64×64 random crops. We also apply random horizontal flips.

BAR. We apply random resized crops using a scale range of 0.7 to 1.0 and selecting aspect ratios between 1.0 to $\frac{4}{3}$. We also apply random horizontal flips.

A.11 Model Calibration

In Fig. A8 and A9 we show the reliability diagrams for ERM model (leftmost column) and for each exit ($E1 - E3$) for OccamResNet for COCO-on-Places and Biased MNIST respectively. In terms of model calibration, OccamNet reduces the expected calibration error (ECE) to some extent, yet there is a large room for improvement, which is an interesting direction to pursue.

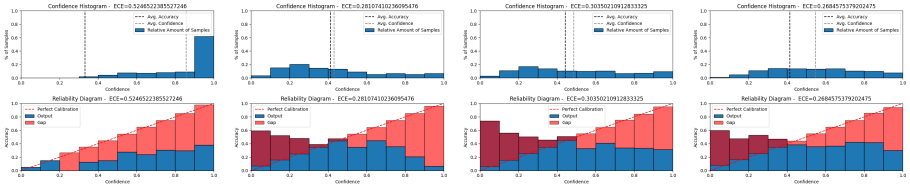

Fig. A8: Reliability diagrams for the classifier trained with ERM (leftmost column) versus exit gate calibrations for $E1 - E3$ (right hand columns) on COCO-on-Places (unbiased backgrounds test split).

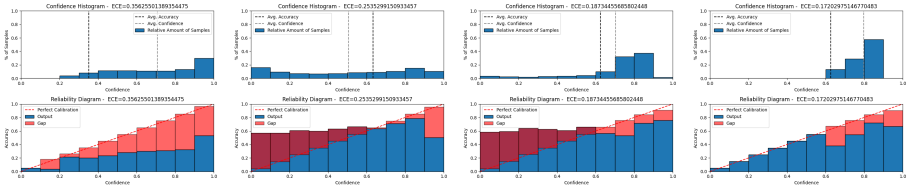

Fig. A9: Reliability diagrams for the classifier trained with ERM (leftmost column) versus exit gate calibrations for $E1 - E3$ (right hand columns) on Biased MNIST.

A.12 Other Architectures and Tasks.

We tested OccamNets implemented with CNNs; however, they may be beneficial to other architectures as well. The ability to exit dynamically could be used with transformers, graph neural networks, and feed-forward networks more generally. There is some evidence already for this on natural language inference tasks, where early exits improved robustness in a transformer architecture [79]. It would be interesting to evaluate multiple existing early exit mechanisms [59] for their abilities to discard spurious correlations. Furthermore, adapting the early exit ideas to non-classification tasks e.g., regression may require small changes e.g., exiting based on continuous error, which can be explored in future works.