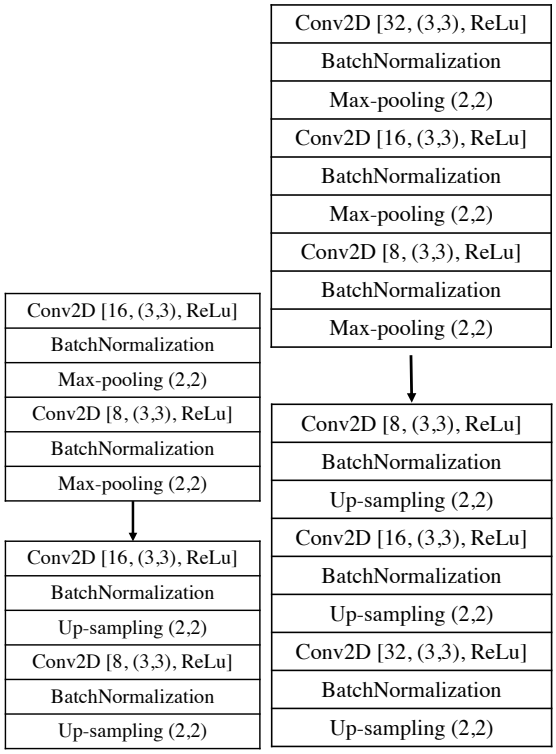
Appendix

A. Additional Experimental Details

A.1. Architectures

TN Architectures. In Figure 4, we show the detailed TN architectures used for the Colored MNIST experiment, as well as for the other models. Architecture details for the downstream classifiers that are specific to each task can be found in Section 4.



(a) Colored MNIST (b) Other datasets

Figure 4. a) Architecture of the TN used for all experiments in the paper.

VIB Architectures. We tested 3 different approaches where we: (a) apply KL regularization on the last layer of the encoder which is of size (1, 2048) and then compute the cross entropy loss; (b) apply KL regularization on the last layer directly, but add 3 fully connected layers of `(1024, ReLU, batch normalization)`, `(512, ReLU, batch normalization)`, and `(256, ReLU)`, before calculating the cross-entropy loss; and (c) follow variant (a) after adding a fully connected layer of size 512 after the last layer of the encoder. We found that (c) yielded the best performance.

A.2. Hyperparameter Configurations

For all the methods including the baselines we have tuned the hyper-parameters such as the optimizer (SGD and Adam), learning rate (0.01, 0.001. 0.0001), and batch size (16, 32, 64). The input size for all the experiments was set to $224 \times 224 \times 3$ except for Colored MNIST, which was $28 \times 28 \times 3$. For Colored MNIST and CelebA, we found that the Adam optimizer with a learning rate of 0.0001 and batch size of 64 worked best. For the Waterbirds and VLCS datasets, we found that the SGD optimizer with learning rate of 0.001 and momentum of 0.9 with batch size of 32 yielded the best performance. For the background challenge, we set the optimizer to SGD with learning rate of 0.001 and batch size to 32.

For CIM and VIB-based approaches, we tested β (the hyperparameter for VIB) and λ (for the contrastive loss) with a range of values within $[1.0, 0.00001]$. In Table 6, we report the best-performing hyperparameters for each method.

Dataset	Method	VIB's β	CIM's λ
CelebA	CIM	-	0.00001
	VIB	0.1	-
	CIM+VIB	0.001	0.0001
Waterbirds	CIM	-	0.0001
	VIB	0.001	-
	CIM+VIB	0.001	0.0001
Background challenge	CIM+VIB	0.001	0.0001
Color MNIST	CIM	-	0.00001
	VIB	0.00001	-
	CIM+VIB	0.00001	0.00001

Table 6. β and λ values for CIM, VIB, and CIM+VIB.

B. Additional Experimental Results

B.1. Out-of-domain Generalization

We present additional results from Section 4.2 with standard errors calculated over 3 runs for our methods. We note that the results from other papers did not include replicates.

Method	Caltech	LabelMe	Pascal	Sun	Average
DeepC (Li et al., 2018b)	87.47	62.06	64.93	61.51	68.89
CIDDG (Li et al., 2018b)	88.83	63.06	64.38	62.10	69.59
CCSA (Motiian et al., 2017)	92.30	62.10	67.10	59.10	70.15
SLRC (Ding & Fu, 2017)	92.76	62.34	65.25	63.54	70.15
TF (Li et al., 2017a)	93.63	63.49	69.99	61.32	72.11
MMD-AAE (Li et al., 2018a)	94.40	62.60	67.70	64.40	72.28
D-SAM (D'Innocente & Caputo, 2018)	91.75	57.95	58.59	60.84	67.03
Shape Bias (Asadi et al., 2019)	98.11	63.61	74.33	67.11	75.79
VIB (Alemi et al., 2016)	97.44±0.143	66.41±0.045	73.29±0.040	68.49±0.150	76.41±0.095
SCL_{E2E} (Ours)	95.56±0.141	66.72±0.043	73.16±0.053	65.10±0.071	75.14±0.077
CIM (Ours)	98.21±0.004	**67.80±0.010**	73.97±0.003	69.01±0.003	77.25±0.005
CIM + VIB (Ours)	**98.81±0.003**	66.49±0.004	**74.89±0.007**	**70.13±0.008**	**77.58±0.006**

Table 7. Multi-source domain generalization results (%) on the VLCS dataset with ResNet-18 as the base network for downstream classification. All reported numbers are averaged over three runs. CIM+VIB outperforms the state-of-the-art model (Asadi et al., 2019).

B.2. Additional Variants of CIM on Subgroup Performance

We present additional results on the other variants of CIM as mentioned in Section 3.2: CIM_{f} and CIM_{g}. We also experiment with a naive metric based on the ℓ_2 distance between two images in pixel space, which we name CIM_{mse}, and find that it

leads to worse performance. We report the results of both these variants on Waterbirds and CelebA datsets in Table 8. We find that encoding only the input (i.e. CIM+VIB) to calculate the structural triplet loss outperforms both CIM_g and CIM_f.

Dataset	Method	unsupervised (group-level),	Worst-group acc.	Average acc.
CelebA	GDRO (Sagawa et al., 2019)	✗	**88.30**	91.80
	ERM (Sagawa et al., 2019)	✓	41.10	94.80
	Our baseline	✓	70.31	93.98
	VIB (Alemi et al., 2016)	✓	78.13	91.94
	SCL_E2E (Ours)	✓	68.80	**95.80**
	CIM_f + VIB (Ours)	✓	80.87	88.24
	CIM_g + VIB (Ours)	✓	82.03	91.27
	CIM + VIB (Ours)	✓	<u>83.59</u>	90.61
Waterbirds	GDRO (Sagawa et al., 2019)	✗	83.80	89.40
	CAMEL (Goel et al., 2020)	✗	**89.70**	90.90
	ERM (Sagawa et al., 2019)	✓	60.00	**97.30**
	Our baseline	✓	62.19	96.42
	VIB (Alemi et al., 2016)	✓	75.31	95.39
	SCL_E2E (Ours)	✓	64.10	96.50
	CIM_f + VIB (Ours)	✓	76.65	95.31
	CIM_g + VIB (Ours)	✓	73.79	94.91
	CIM + VIB (Ours)	✓	<u>77.23</u>	95.60

Table 8. Average and worst-group accuracies for CelebA and Waterbird benchmark datasets. Methods without group-level supervision (✓) are preferable over those with group-level supervision (✗). CIMs + VIB outperforms unsupervised methods on both datasets, while achieving comparable performance against supervised approaches. Underline shows the best accuracy among the unsupervised methods.

B.3. Connection to Weighted attention and Saliency Maps

Although our method bears resemblance to methods with (weighted) attention and saliency, we demonstrate that the TN in CIM is in fact learning a more sophisticated transformation. We experimented with two approaches: (1) learning an attention-like map (`Only m`) via the TN; and (2) GradCAM (Selvaraju et. al 2016) for saliency detection (`Saliency`) to see which method would yield perform more favorably on the downstream classification task.

For the GradCAM baseline, we first trained a ResNet-50 classifier, computed saliency maps using GradCAM after freezing the model, composed the saliency map with the inputs, and trained a new ResNet-50 classifier with the modified inputs (without nuisance background information). As in Table 9, we find that this saliency method (`Saliency`) improves over the baseline, but does not outperform either the learned attention map (`Only m`) or CIM. Figure 5 shows that this is because the saliency detection algorithm may incorrectly mask out a task-relevant region in the original image. By learning this transformation *jointly* with our classifier as in CIM, we are able to improve performance.

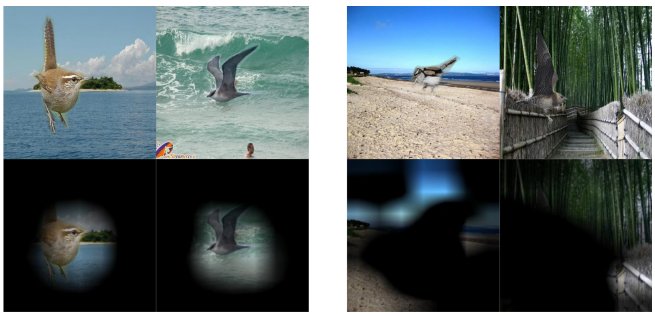


Figure 5. Learned saliency maps from the GradCAM baseline.

B.4. ALTERNATIVE SAMPLING STRATEGIES FOR CIM

Inspired by the contrastive learning literature which explores the impact of various positive and negative sampling strategies, we experimented with two alternative sampling approaches: `neg++` and `pos++` (sampling more negative/positive examples per batch respectively). In `neg++` we used 3 negative examples for each positive example, and vice versa. As shown in Table 9, we did not find a clear improvement.

	CelebA		Waterbirds	
Method	**Worst-group**	**Average**	**Worst-group**	**Average**
Baseline	70.31	**93.91**	62.19	**96.42**
Only m	82.81	90.06	75.31	95.14
Saliency	75.77	90.90	67.19	92.70
CIM_{mse}	75.78	91.55	69.69	95.46
CIM neg++	78.12	91.90	70.62	96.10
CIM pos++	78.13	92.89	76.71	95.60
CIM	**83.59**	90.61	**77.23**	95.60

Table 9. Results for different baselines and sampling strategies.