

Appendix

A. More Details of Datasets

A.1. UrbanCars Details

Here we present more details of the UrbanCars dataset.

Number of Images Regarding the number of images, each target class contains 4000 images in the training set, *i.e.*, 8000 images in total. That is, our training set is balanced regarding the target label and only imbalanced with shortcut labels. Therefore, UrbanCars does not have a target class imbalance issue [39] in Waterbirds dataset [74], where 76.8% of images are waterbird, and 23.3% of images are landbird. In validation and testing sets of UrbanCars, each split contains 500 images.

Data Annotation As mentioned in Sec. 2.1, each image is annotated with three image-level labels—car body type, background, and co-occurring object. Besides, following Waterbirds [74] dataset, the dataset also contains the mask annotation of the car object and the co-occurring object, which enables shortcut mitigation via targeted augmentation (category 2), *i.e.*, CF+F Aug [11] (*cf.* Appendix B.4) and our proposed LLE approach (*cf.* Appendix B.5).

Details of Data Construction Here, we present the details of collecting the data from source datasets based on three visual cues—main object (*i.e.*, car), background shortcut, and co-occurring object shortcut.

First, to obtain car images, we use MaskFormer [14] pretrained on MS-COCO [57] dataset’s panoptic segmentation [47] task to segment cars from Stanford Cars [50] dataset. In each image from Stanford Cars, we choose the predicted car instance mask that has the largest IoU with the bounding box annotation provided in Stanford Cars. After segmentation, we run MaskFormer on foreground-only images to detect humans. Images with humans detected are filtered out.

When pasting the car object to the background, we first compute its square bounding box, which is the bounding box whose side length is the longer side of the actual bounding box of the car object based on the predicted segmentation mask. Then, we resize the square bounding box such that the side length is 50% of the final image size, which is smaller than the size of the car object.

We merge the original 196 classes in Stanford Cars into urban cars (*e.g.*, sedan, hatchback, *etc.*) and country cars (*e.g.*, pickup truck, van, *etc.*). The mapping from the original 196 classes in Stanford Cars to *urban cars* and *country cars* is as follows:

- *urban cars*: Acura RL Sedan 2012, Acura TL Sedan 2012, Acura TL Type-S 2008, Acura TSX Sedan 2012, Acura Integra Type R 2001, Acura ZDX Hatchback 2012, Aston Martin V8 Vantage Coupe 2012, Aston Martin Virage Convertible 2012, Aston Martin Virage Coupe 2012, Audi RS 4 Convertible 2008, Audi A5 Coupe 2012, Audi TTS Coupe 2012, Audi R8 Coupe 2012, Audi V8 Sedan 1994, Audi 100 Sedan 1994, Audi 100 Wagon 1994, Audi TT Hatchback 2011, Audi S6 Sedan 2011, Audi S5 Convertible 2012, Audi S5 Coupe 2012, Audi S4 Sedan 2012, Audi S4 Sedan 2007, Audi TT RS Coupe 2012, BMW ActiveHybrid 5 Sedan 2012, BMW 1 Series Convertible 2012, BMW 1 Series Coupe 2012, BMW 3 Series Sedan 2012, BMW 3 Series Wagon 2012, BMW 6 Series Convertible 2007, BMW M3 Coupe 2012, BMW M5 Sedan 2010, BMW M6 Convertible 2010, BMW Z4 Convertible 2012, Bentley Continental Supersports Conv. Convertible 2012, Bentley Arnage Sedan 2009, Bentley Mulsanne Sedan 2011, Bentley Continental GT Coupe 2012, Bentley Continental GT Coupe 2007, Bentley Continental Flying Spur Sedan 2007, Bugatti Veyron 16.4 Convertible 2009, Bugatti Veyron 16.4 Coupe 2009, Buick Regal GS 2012, Buick Verano Sedan 2012, Cadillac CTS-V Sedan 2012, Chevrolet Corvette Convertible 2012, Chevrolet Corvette ZR1 2012, Chevrolet Corvette Ron Fellows Edition Z06 2007, Chevrolet Camaro Convertible 2012, Chevrolet Impala Sedan 2007, Chevrolet Sonic Sedan 2012, Chevrolet Cobalt SS 2010, Chevrolet Malibu Hybrid Sedan 2010, Chevrolet Monte Carlo Coupe 2007, Chevrolet Malibu Sedan 2007, Chrysler Sebring Convertible 2010, Chrysler 300 SRT-8 2010, Chrysler Crossfire Convertible 2008, Chrysler PT Cruiser Convertible 2008, Daewoo Nubira Wagon 2002, Dodge Caliber Wagon 2012, Dodge Caliber Wagon 2007, Dodge Magnum Wagon 2008, Dodge Challenger SRT8 2011, Dodge Charger Sedan 2012, Dodge Charger SRT-8 2009, Eagle Talon Hatchback 1998, FIAT 500 Abarth 2012, FIAT 500 Convertible 2012, Ferrari FF Coupe 2012, Ferrari California Convertible 2012, Ferrari 458 Italia Convertible 2012, Ferrari 458 Italia Coupe 2012, Fisker Karma Sedan 2012, Ford Mustang Convertible 2007, Ford GT Coupe 2006, Ford Focus Sedan 2007, Ford Fiesta Sedan 2012, Geo Metro Convertible 1993, Honda Accord Coupe 2012, Honda Accord Sedan 2012, Hyundai Veloster Hatchback 2012, Hyundai Sonata Hybrid Sedan 2012, Hyundai Elantra Sedan 2007, Hyundai Accent Sedan 2012, Hyundai Genesis Sedan 2012, Hyundai Sonata Sedan 2012, Hyundai Elantra Touring Hatchback 2012, Hyundai Azera Sedan 2012, Infiniti G Coupe IPL

2012, Jaguar XK XKR 2012, Lamborghini Reventon Coupe 2008, Lamborghini Aventador Coupe 2012, Lamborghini Gallardo LP 570-4 Superleggera 2012, Lamborghini Diablo Coupe 2001, Lincoln Town Car Sedan 2011, MINI Cooper Roadster Convertible 2012, Maybach Landaulet Convertible 2012, McLaren MP4-12C Coupe 2012, Mercedes-Benz 300-Class Convertible 1993, Mercedes-Benz C-Class Sedan 2012, Mercedes-Benz SL-Class Coupe 2009, Mercedes-Benz E-Class Sedan 2012, Mercedes-Benz S-Class Sedan 2012, Mitsubishi Lancer Sedan 2012, Nissan Leaf Hatchback 2012, Nissan Juke Hatchback 2012, Nissan 240SX Coupe 1998, Plymouth Neon Coupe 1999, Porsche Panamera Sedan 2012, Rolls-Royce Phantom Drophead Coupe Convertible 2012, Rolls-Royce Ghost Sedan 2012, Rolls-Royce Phantom Sedan 2012, Scion xD Hatchback 2012, Spyker C8 Convertible 2009, Spyker C8 Coupe 2009, Suzuki Aerio Sedan 2007, Suzuki Kizashi Sedan 2012, Suzuki SX4 Hatchback 2012, Suzuki SX4 Sedan 2012, Tesla Model S Sedan 2012, Toyota Camry Sedan 2012, Toyota Corolla Sedan 2012, Volkswagen Golf Hatchback 2012, Volkswagen Golf Hatchback 1991, Volkswagen Beetle Hatchback 2012, Volvo C30 Hatchback 2012, Volvo 240 Sedan 1993, smart fortwo Convertible 2012.

- *country* cars: AM General Hummer SUV 2000, Aston Martin V8 Vantage Convertible 2012, BMW X5 SUV 2007, BMW X6 SUV 2012, BMW X3 SUV 2012, Buick Rainier SUV 2007, Buick Enclave SUV 2012, Cadillac SRX SUV 2012, Cadillac Escalade EXT Crew Cab 2007, Chevrolet Silverado 1500 Hybrid Crew Cab 2012, Chevrolet Traverse SUV 2012, Chevrolet HHR SS 2010, Chevrolet Tahoe Hybrid SUV 2012, Chevrolet Express Cargo Van 2007, Chevrolet Avalanche Crew Cab 2012, Chevrolet TrailBlazer SS 2009, Chevrolet Silverado 2500HD Regular Cab 2012, Chevrolet Silverado 1500 Classic Extended Cab 2007, Chevrolet Express Van 2007, Chevrolet Silverado 1500 Extended Cab 2012, Chevrolet Silverado 1500 Regular Cab 2012, Chrysler Aspen SUV 2009, Chrysler Town and Country Minivan 2012, Dodge Caravan Minivan 1997, Dodge Ram Pickup 3500 Crew Cab 2010, Dodge Ram Pickup 3500 Quad Cab 2009, Dodge Sprinter Cargo Van 2009, Dodge Journey SUV 2012, Dodge Dakota Crew Cab 2010, Dodge Dakota Club Cab 2007, Dodge Durango SUV 2012, Dodge Durango SUV 2007, Ford F-450 Super Duty Crew Cab 2012, Ford Freestar Minivan 2007, Ford Expedition EL SUV 2009, Ford Edge SUV 2012, Ford Ranger SuperCab 2011, Ford F-150 Regular Cab 2012, Ford F-150 Regular Cab 2007, Ford E-Series Wagon Van 2012, GMC Terrain SUV 2012, GMC Savana Van 2012, GMC Yukon Hybrid SUV 2012, GMC Acadia SUV 2012, GMC Canyon Extended Cab 2012, HUMMER H3T Crew Cab 2010, HUMMER H2 SUT Crew Cab 2009, Honda Odyssey Minivan 2012, Honda Odyssey Minivan 2007, Hyundai Santa Fe SUV 2012, Hyundai Tucson SUV 2012, Hyundai Veracruz SUV 2012, Infiniti QX56 SUV 2011, Isuzu Ascender SUV 2008, Jeep Patriot SUV 2012, Jeep Wrangler SUV 2012, Jeep Liberty SUV 2012, Jeep Grand Cherokee SUV 2012, Jeep Compass SUV 2012, Land Rover Range Rover SUV 2012, Land Rover LR2 SUV 2012, Mazda Tribute SUV 2011, Mercedes-Benz Sprinter Van 2012, Nissan NV Passenger Van 2012, Ram C/V Cargo Van Minivan 2012, Toyota Sequoia SUV 2012, Toyota 4Runner SUV 2012, Volvo XC90 SUV 2007.

Second, regarding the background images for the background shortcut, we use images from the Places [99] dataset, where the *urban* background images are from alley, crosswalk, downtown, gas station, garage (outdoor), driveway classes, and the *country* background images are forest road, field road, desert road. We use MaskFormer mentioned above to detect humans, cars, and co-occurring objects (*e.g.*, fireplug) on Places images. Images with the aforementioned object categories detected will be filtered out. When used as the background image in UrbanCars, we resize each image to 256×256 .

Lastly, the co-occurring objects are from LVIS [29] based on its ground-truth instance segmentation mask, where *urban* co-occurring object images are from fireplug, stop sign, street sign, parking meter, traffic light and *country* co-occurring object images are from farm animals—cow, horse, sheep. We filter out instance masks with more than one connected component (*e.g.*, instances with more than one connected component are usually occluded by other objects). When pasting to the background, the square bounding box (see above) of the co-occurring object is resized such that the side length is 25% of the final image size.

Dataset Release Since Places dataset (the source dataset for the background) does not own the copyright of images, we cannot directly release the final images in UrbanCars. Instead, we release the code that creates the UrbanCars from source datasets.

A.2. ImageNet-Watermark (ImageNet-W) Details

Here we show more details about creating the ImageNet-Watermark dataset. Regarding the position, we paste the watermark at the center of the image. More specifically, the XY-position of the top-left corner of the watermark is $(0.01W, 0.4H)$, where W and H are the width and height of the input image for the models. Regarding the font size, we use 36 for the 224×224 sized images, which is the most common input size for most vision models. For large foundation models using larger input sizes, we use 62, 82, 84 for 384×384 , 512×512 , 518×518 sized images, respectively, where the font sizes are approximately 0.16 times smaller to the image size. The font color for the watermark is (255, 255, 255, 128) in RGBA, which is a transparent

white color. We use the open-sourced “SourceHanSerifSC-ExtraLight”[†] as the font family.

Content of Watermark As mentioned in Sec. 2.2, we use “捷径捷径捷径” as the content of the watermark. We show the results of using other contents or languages in Tab. 8. When using other content in Simplified Chinese (*e.g.*, “一二三四五六”) or other languages (*i.e.*, Japanese, Korean, English, and Arabic), we observe smaller IN-W Gap and Carton Gap. We conjecture this is due to the simpler shape of other contents compared to “捷径捷径捷径” used in the ImageNet-W. Nevertheless, the accuracy drops across different contents suggest that it is the presence of the watermark rather than its content that causes the watermark shortcut reliance. Besides, the watermark shortcut reliance is stronger when the watermark’s content looks more visually similar to the pattern of the watermark in carton class images in ImageNet-1k training set, *e.g.*, Simplified Chinese characters with complex shapes (*cf.* Fig. 7).







watermark content	language	English translation	Example Image	IN-W Gap	Carton Gap
捷径捷径捷径	Simplified Chinese	shortcut shortcut shortcut		-26.64	+40
一二三四五六	Simplified Chinese	one two three four five six		-6.12	+22
ショートカット	Japanese	shortcut		-2.66	+18
지름길지름길	Korean	shortcut shortcut		-12.30	+34
shortcut	English	N/A		-6.39	+8
abcdefghijkl	English	N/A		-5.54	+4
الاختصار	Arabic	shortcut		-7.79	+4

Table 8. Ablation study on ResNet-50’s reliance on the watermark shortcut in different content and languages. Watermarks of various contents can cause shortcut reliance. We choose the content shown in the first row for creating ImageNet-W as it causes a larger IN-W Gap and Carton Gap and is more visually similar to the watermark that appears in the IN-1k training set.

[†]<https://source.typekit.com/source-han-serif/>

Dataset Release We release the code of adding watermarks instead of directly releasing the final images. We follow AugLy [63] to implement the code of adding watermarks, which is encapsulated as a function similar to PyTorch’s transforms API. It is easy to use and can evaluate vision models on the fly by simply adding the watermark transform function with ImageNet-1k validation set downloaded, *i.e.*, no need to save images with watermarks to the disk in advance.

B. Implementation Details

Here we present more details of the benchmark methods and our Last Layer Ensemble approach.

B.1. Watermark Augmentation (WMK Aug)

To mitigate the watermark shortcut on ImageNet, we propose simple-yet-effective watermark augmentation (WMK Aug). Concretely, we overlay a random watermark onto the training images in ImageNet-1k. The watermark is random in terms of (1) position, (2) font size, and (3) content, where we use random CJK (Chinese, Japanese, and Korean) characters in a random number of characters. The randomness of watermark augmentation in training avoids being identical to the watermark used for evaluation on ImageNet-W.

B.2. Background Augmentation (BG Aug)

To mitigate the background shortcut on ImageNet, we follow [73,93] and use background augmentation (BG Aug). Concretely, we use unsupervised saliency segmentation developed by Ryali *et al.* [73] to separate the foreground object from the backgrounds in each image. Then “tiled” background images are created by repeating the procedure of pasting the largest rectangular of the background onto the foreground region to cover the foreground object (more details in [93]). Finally, to augment the background, we paste the segmented foreground object from class A onto a tiled background from class B ($A \neq B$).

B.3. Detailed Experiment Settings

UrbanCars On UrbanCars, we follow the standard regularization setting in [74]. Concretely, we use stochastic gradient descent (SGD) optimizer with 10^{-3} learning rate and 10^{-4} weight decay (*i.e.*, ℓ_2 penalty). We use 128 for the batch size. All models are trained with 300 epochs, and we use the early stopped epoch that achieves the best validation set worst-group accuracy to report the final results on the testing set. Specifically, for methods that do not use ground-truth shortcut labels (*i.e.*, category 1, 2, 4), the worst-group accuracy is computed based on labels of both shortcuts, *i.e.*, lowest accuracy among all eight groups. Methods using shortcut labels (*i.e.*, category 3) may encounter the issue in which one or a subset of shortcuts remain unlabeled or even unknown. To simulate the situation, besides standard setting using labels of both shortcuts, we additionally create two settings—(1) only using BG label; (2) only using CoObj label (*cf.* bottom two sections in Tab. 5). In both cases, the worst-group accuracy on the validation set also only considers the label of one shortcut, *i.e.*, the lowest accuracy among four groups based on the combination of the target label and the single shortcut label. Each experiment on UrbanCars is repeated six times using different random seeds, and we report the average results over six runs.

ImageNet On ImageNet, we use last layer re-training [46] to only train the last classification layer upon a frozen feature extractor to benchmark methods in Tab. 4 and our Last Layer Ensemble (LLE) method in Tab. 6. Note that we directly evaluate self-supervised approaches and foundation models in Tab. 6 without using last layer re-training. When using ResNet-50 network architecture with last layer re-training (*i.e.*, methods in Tab. 4), we use SGD optimizer with 10^{-4} weight decay. For all models, we tune the learning rate over $\{10^{-2}, 10^{-3}, 10^{-4}\}$ and choose the one with the best top-1 accuracy on IN-1k. We use 1024 for the batch size. Unlike the detailed implementation in [46], we do not train the last classification layer from scratch but initialize it by the weights of ERM’s last layer because we find that the latter way converges faster. Note that ERM’s last layer is also re-trained (*e.g.*, ERM in Tab. 4). When applying our LLE approach with the MAE feature extractor, we follow MAE [30] to use 0 weight decay.

B.4. Details of Benchmark Methods

We introduce more details (*e.g.*, hyperparameters) of benchmark methods in each category.

Category 1: Standard Augmentation and Regularization Following PyTorch’s new training recipe [86], we use $\alpha = 0.2$ for Mixup, $p = 0.1$ for Cutout, and $\alpha = 1.0$ for CutMix on both UrbanCars and ImageNet experiments. For AugMix, we use all default hyperparameters in the original implementation. For the co-efficient of ℓ_2 penalty of logits in SD, we use 0.1 on UrbanCars and 10^{-4} on ImageNet (we find that SD using 0.1 on ImageNet achieves poor results).

Category 2: Targeted Augmentation for Mitigating Shortcuts For CF+F Aug [11], based on the ground-truth masks (*cf.* Appendix A.1), we use CF(Grey) and F(Random) for generating counterfactual and factual augmentations because they achieve the best results on Waterbirds when not using external generative models. Concretely, CF(Grey) infills the grey color to the bounding box area of the object to generate the counterfactual image, and F(Random) uses random noises to replace the background area—outside of the bounding box of the car object (more details in [11]).

For style transfer [27] (*i.e.*, texture augmentation or TXT Aug), we use the official code to generate Stylized ImageNet (SIN) for training. The details of BG Aug and WTM Aug are introduced in Appendix B.2 and Appendix B.1, respectively. Note that WTM Aug, TXT Aug, and BG Aug shown in Tab. 4 jointly use augmented images and original IN-1k images for training.

Category 3: Using Shortcut Labels We follow the original GroupDRO (gDRO)’s implementation to use 0.01 step size and $\gamma = 0.1$. For Domain Independent (DI), its number of domains is decided based on the usage of shortcut labels, *i.e.*, 2 when using labels of only one shortcut and 4 for using labels of both shortcuts. We follow SUBG’s implementation to subsample the training data to rebalance the data, where each group has (fewer but) the same number of images. For DFR, we use its DFR_{Tr}^{Tr} variant where ERM’s last layer is re-trained on a balanced sub-sampled training set (*i.e.*, SUBG).

Category 4: Inferring Pseudo Shortcut Labels For LfF, we follow the original implementation to set $q = 0.7$. As discussed in Sec. 5.2, JTT and EILL use an early-stopped ERM as the reference model to infer the pseudo shortcut labels, where we use E to denote the number of training epochs of the reference ERM model. For JTT, we use E=1 and E=2 on UrbanCars. Since JTT [59] use E=40,50,60 on Waterbirds, we also show their results on UrbanCars in Appendix D.1. We use $\lambda_{up} = 100$ for JTT on UrbanCars. On ImageNet, we use E=1 and $\lambda_{up} = 5$ because we found $\lambda_{up} = 100$ (*i.e.*, sampling wrongly predicted examples 100 times) is not scalable on the larger ImageNet dataset. For EILL, we use E=1 and E=2 on UrbanCars and E=1 on ImageNet. We use gDRO as the invariant learner for EILL (more details in [15]). While DebiAN uses a full network as the shortcut “discoverer” (more details in [54]), we use a single fully-connected layer on top of the feature extractor for its experiments on ImageNet under the last layer re-training setting.

B.5. Details of Last Layer Ensemble (LLE)

On UrbanCars, we augment background and co-occurring object visual cues to mitigate multiple shortcuts based on ground-truth masks (*cf.* Appendix A.1). Concretely, we use ground-truth masks of the car object and co-occurring object to (1) segment car object; (2) segment co-occurring object; (3) create the tiled background, a background-only image where the regions of the object and co-occurring object are tiled (*cf.*, Appendix B.2). To augment the background, we sample segmented car object and co-occurring object from class A and tiled background from class B ($A \neq B$), which are used to form the background-augmented images—pasting car object and co-occurring object on the tiled background. Similarly, to augment the co-occurring object, we sample the segmented car object and tiled background from class A and sample the segmented co-occurring object from class B ($A \neq B$) to create the augmented images. Note that we only use the target label of the car body type for augmentation. In other words, neither the BG shortcut labels nor the CoObj shortcut labels are used. After obtaining two types of augmented images, LLE uses three last classification layers as an ensemble—two layers for two shortcuts and one layer for the original images. The distributional shift classifier predicts three shift categories: (1) no shift (*i.e.*, original images), (2) background shift (*i.e.*, background-augmented images), (3) co-occurring object shift (*i.e.*, co-occurring object augmented images).

On ImageNet, LLE uses style transfer [27] (*i.e.*, TXT Aug) to mitigate the texture shortcut, BG Aug (details in Appendix B.2) to mitigate the background shortcut, and WMK Aug (details in Appendix B.1) to mitigate the watermark shortcut. LLE jointly trains four last classification layers as an ensemble—three layers for three shortcuts and one layer for original images in IN-1k. The distributional shift classifier predicts four categories: (1) no shift (original images from IN-1k), (2) texture shift (*i.e.*, texture augmented images), (3) background shift (*i.e.*, background augmented images), (4) watermark shift (*i.e.*, watermark augmented images).

C. Results of LfMF (Extended version of LfF)

One may suggest that the Whac-A-Mole problem in multi-shortcut mitigation can be solved by straightforwardly extending existing approaches designed for single-shortcut mitigation. To this end, we extend the Learning from Failure (LfF) [61] method. The original LfF method trains two networks—a bias-amplified network to identify shortcuts and a debiased network to mitigate the identified shortcuts. We extend LfF by adding the second bias-amplified network, where we investigate whether

	I.D. Acc	BG Gap	CoObj Gap	BG+CoObj Gap
ERM	97.6	-15.3	-11.2	-69.2
LfMF	97.7	-15.6 ($\times 1.02$)	-12.7 ($\times 1.13$)	-71.2

Table 9. Results of LfMF on UrbanCars dataset.

	I.D. Acc	shortcut reliance		
		BG Gap	CoObj Gap	BG+CoObj Gap
ERM	97.6	-15.3	-11.2	-69.2
JTT (E=1)	95.9	-8.1	-13.3 ($\times 1.18$)	-37.6
JTT (E=2)	94.6	-23.3 ($\times 1.52$)	-5.3	-52.1
JTT (E=40)	97.7	-15.8 ($\times 1.03$)	-10.7	-69.3
JTT (E=50)	97.6	-14.8	-11.0	-67.9
JTT (E=60)	97.2	-15.1	-10.7	-70.5

Table 11. Results of JTT when using ERM trained with other epochs ($E \in \{40, 50, 60\}$) as the reference model to infer pseudo shortcut labels on UrbanCars.

	IN-1k	watermark		texture		background
		IN-W Gap	Carton Gap	SIN Gap	IN-R Gap	IN-9 Gap
ERM	76.39	-25.40	+30	-69.43	-56.22	-5.19
LfMF	76.38	-26.95 ($\times 1.06$)	+32 ($\times 1.06$)	-69.29	-55.93	-5.70 ($\times 1.10$)

Table 10. Results of LfMF on ImageNet.

	I.D. Acc	Shortcut Reliance		
		BG Gap	CoObj Gap	BG+CoObj Gap
ERM	97.6	-15.3	-11.2	-69.2
w/o stop gradient	97.3	-3.3	-2.6	-7.7
w/ frozen feature extractor	97.3	-11.2	-9.0	-43.3
LLE	96.7	-2.1	-2.7	-5.9

Table 12. Results of ablation study of LLE on UrbanCars dataset.

two bias-amplified networks can identify different shortcuts for mitigation. We name this method *Learning from Multiple Failures* (LfMF). The results in Tabs. 9 and 10 show that LfMF still amplifies shortcuts over ERM, demonstrating that a simple extension of existing methods cannot easily solve the Whac-A-Mole problem.

D. More Results on UrbanCars

D.1. More Results of JTT

In Sec. 5.2, we show the result of JTT when $E=1$ and $E=2$. Since JTT tunes E over $\{40, 50, 60\}$ on Waterbirds [74] (more epochs for training the reference ERM models to infer pseudo shortcut labels). Here, we also show the results of JTT when $E \in \{40, 50, 60\}$ in Tab. 11, where JTT either exhibits Whac-A-Mole results by amplifying the background shortcut or barely mitigates either shortcut compared to ERM.

D.2. Ablation Study of LLE on UrbanCars

As mentioned in Sec. 4, when training the distributional shift classifier, we stop the gradient from the distributional shift classifier to the feature extractor under the end-to-end training setting on UrbanCars. Here we show the ablation study in Tab. 12, where the variant without stopping the gradient achieves suboptimal results. The results demonstrate the necessity of stopping the gradient to prevent the feature extractor from learning the shortcut information used in the distributional shift classifier’s supervision.

While we use the end-to-end training setting for experiments on UrbanCars, we also show the results of LLE with the last layer re-training setting (*cf.* frozen feature extractor in Tab. 12), which shows that using a frozen feature extractor can also improve the results over ERM, but the results are also suboptimal compared to end-to-end training.

E. More Results on ImageNet-W

E.1. Results of More Methods on ImageNet-W

The results of more methods in addition to methods in Tab. 1 are shown in Tab. 13. We observe a pervasive watermark shortcut reliance across network architecture, pretraining datasets, supervision, mitigation methods, *etc.*

E.2. ImageNetV2-W: ImageNet-W with ImageNetV2

To further verify the pervasiveness of watermark shortcut reliance, we also overlay the watermark on ImageNetV2 [69] dataset to construct the ImageNet-W test set. We denote this ImageNet-W variant as **ImageNetV2-W**. The results are shown in Tab. 14, which is comparable to results on ImageNet-W shown in Tab. 1. Note that some models show +0 Carton Gap results (*e.g.*, CLIP pretrained on WIT and LAION-400M). We conjecture that it is due to the small number (*i.e.*, ten) of carton class images in ImageNetV2. Nevertheless, they still show a considerable predicted probability increase of carton class images ($\Delta P(\hat{y} = \text{carton} \mid y = \text{carton})$). Therefore, the results on ImageNetV2-W strengthen our claim of the watermark shortcut for predicting the carton class learned by various vision models.

method	architecture	(pre)training data	IN-1k Acc \uparrow	$P(\hat{y} = \text{carton})$ (%)	IN-W Gap \uparrow	$\Delta P(\hat{y} = \text{carton})$ (%) \downarrow	Carton Gap \downarrow	$\Delta P(\hat{y} = \text{carton} y = \text{carton})$ (%) \downarrow
Supervised	ResNet-50 [31]	IN-1k [18]	76.1	0.07	-26.7	+7.56	+40	+42.46
MoCov3 (LP)	ResNet-50	IN-1k	74.6	0.08	-20.7	+2.94	+44	+44.37
Style Transfer [27]	ResNet-50	SIN [27]	60.1	0.10	-17.3	+4.91	+52	+50.06
Mixup [95]	ResNet-50	IN-1k	76.1	0.07	-18.6	+3.43	+38	+39.78
CutMix [94]	ResNet-50	IN-1k	78.5	0.09	-14.8	+1.92	+22	+29.61
Cutout [20,98]	ResNet-50	IN-1k	77.0	0.08	-18.0	+2.93	+32	+38.06
AugMix [36]	ResNet-50	IN-1k	77.5	0.09	-16.8	+2.61	+36	+34.44
BiT-M [49]	ResNet-50v2 [32]	IN-21k	82.3	0.09	-8.6	+0.60	+28	+29.73
Supervised	RG-32gf	IN-1k	80.8	0.09	-14.1	+3.74	+32	+33.43
SEER [28] (FT)	RG-32gf [68]	IG-1B [28]	83.3	0.09	-6.5	+0.56	+18	+24.26
SWAG [81] (LP)	RG-32gf	IG-3.6B [81]	84.6	0.08	-6.5	+0.36	+22	+20.56
SWAG (FT)	RG-32gf	IG-3.6B	86.8	0.08	-4.5	+0.49	+30	+26.03
Supervised	ViT-B/32 [22]	IN-1k	75.9	0.09	-8.7	+1.20	+34	+34.31
Uniform Soup [91] (FT)	ViT-B/32	WIT [67]	79.9	0.09	-7.9	+0.32	+24	+23.87
Greedy Soup [91] (FT)	ViT-B/32	WIT	81.0	0.09	-6.5	+0.35	+16	+23.87
Supervised	ViT-B/16	IN-1k	81.0	0.08	-6.7	+0.73	+26	+31.28
RobustViT [12]	ViT-B/16	IN-1k	80.3	0.08	-7.3	+0.44	+34	+37.06
MoCov3 (LP)	ViT-B/16	IN-1k	76.6	0.09	-16.0	+1.97	+22	+38.34
MAE (FT)	ViT-B/16	IN-1k	83.7	0.09	-4.6	+0.67	+24	+22.46
SWAG (LP)	ViT-B/16	IG-3.6B	81.8	0.08	-7.7	+0.46	+18	+19.74
SWAG (FT)	ViT-B/16	IG-3.6B	85.2	0.09	-5.4	+0.45	+24	+25.95
Supervised	ViT-L/16	IN-1k	79.6	0.08	-6.2	+0.82	+34	+32.57
MAE (FT)	ViT-L/16	IN-1k	85.9	0.09	-4.4	+0.50	+22	+22.70
SWAG (LP)	ViT-L/16	IG-3.6B	85.1	0.08	-5.7	+0.23	+6	+9.72
SWAG (FT)	ViT-L/16	IG-3.6B	88.0	0.09	-3.2	+0.24	+20	+19.14
CLIP [67] (zero-shot)	ViT-L/14	WIT [67]	76.5	0.06	-4.4	+0.01	+12	+1.75
CLIP (zero-shot)	ViT-L/14	LAION-400M [76]	72.7	0.05	-4.9	+0.03	+12	+13.76
MAE (FT)	ViT-H/14	IN-1k	86.9	0.08	-3.5	+0.43	+30	+29.59
SWAG (LP)	ViT-H/14	IG-3.6B	85.7	0.09	-4.9	+0.19	+8	+12.80
SWAG (FT)	ViT-H/14	IG-3.6B	88.5	0.09	-3.1	+0.35	+18	+20.25
CLIP (zero-shot)	ViT-H/14	LAION-2B [75]	77.9	0.06	-3.6	+0.03	+16	+12.01
CLIP (zero-shot)	ViT-G/14	LAION-2B	76.6	0.06	-3.8	+0.02	+12	+5.61

Table 13. Results of more methods (also include the methods in Tab. 1). LP and FT stand for linear probing and fine-tuning on ImageNet-1k, respectively.

method	architecture	(pre)training data	IN-1k Acc \uparrow	$P(\hat{y} = \text{carton})$ (%)	IN-W Gap \uparrow	$\Delta P(\hat{y} = \text{carton})$ (%) \downarrow	Carton Gap \downarrow	$\Delta P(\hat{y} = \text{carton} y = \text{carton})$ (%) \downarrow
Supervised	ResNet-50	IN-1k	63.19	0.09	-26.07	+9.29	+70	+53.50
MoCov3 (LP)	ResNet-50	IN-1k	61.98	0.09	-19.83	+3.33	+40	+44.43
Style Transfer	ResNet-50	SIN	48.63	0.09	-15.88	+5.16	+40	+40.28
Supervised	RG-32gf	IN-1k	69.67	0.10	-16.59	+5.21	+40	+34.09
SEER (FT)	RG-32gf	IG-1B	72.48	0.08	-9.00	+0.76	+30	+31.03
SWAG (LP)	RG-32gf	IG-3.6B	75.51	0.10	-7.48	+0.45	+20	+17.57
SWAG (FT)	RG-32gf	IG-3.6B	78.18	0.09	-5.67	+0.74	+30	+27.15
Supervised	ViT-B/32	IN-1k	62.99	0.07	-8.45	+1.39	+30	+20.97
Uniform Soup (FT)	ViT-B/32	WIT	68.58	0.08	-8.57	+0.42	+60	+47.84
Greedy Soup (FT)	ViT-B/32	WIT	69.54	0.08	-7.43	+0.44	+50	+40.78
Supervised	ViT-B/16	IN-1k	69.55	0.09	-7.55	+0.92	+40	+22.66
MoCov3 (LP)	ViT-B/16	IN-1k	65.25	0.09	-16.32	+2.40	+50	+41.75
MAE (FT)	ViT-B/16	IN-1k	73.20	0.10	-6.12	+1.05	+50	+38.12
SWAG (LP)	ViT-B/16	IG-3.6B	72.87	0.10	-8.66	+0.55	+10	+20.01
SWAG (FT)	ViT-B/16	IG-3.6B	75.57	0.09	-6.51	+0.66	+40	+32.34
Supervised	ViT-L/16	IN-1k	67.49	0.07	-7.37	+0.99	+30	+37.09
MAE (FT)	ViT-L/16	IN-1k	76.65	0.10	-6.57	+0.87	+40	+33.43
SWAG (LP)	ViT-L/16	IG-3.6B	76.64	0.09	-6.71	+0.30	+30	+12.46
SWAG (FT)	ViT-L/16	IG-3.6B	80.39	0.10	-4.14	+0.36	+20	+30.21
CLIP (zero-shot)	ViT-L/14	WIT	70.87	0.09	-5.29	+0.02	+0	+4.20
CLIP (zero-shot)	ViT-L/14	LAION-400M	65.43	0.06	-5.90	+0.02	+0	+9.44
MAE (FT)	ViT-H/14	IN-1k	78.46	0.10	-5.26	+0.71	+30	+31.43
SWAG (LP)	ViT-H/14	IG-3.6B	77.38	0.10	-6.46	+0.23	+0	+10.74
SWAG (FT)	ViT-H/14	IG-3.6B	81.06	0.09	-4.39	+0.46	+10	+21.45
CLIP (zero-shot)	ViT-H/14	LAION-2B	70.92	0.08	-4.44	+0.02	+30	+19.09
CLIP (zero-shot)	ViT-G/14	LAION-2B	69.65	0.09	-5.16	+0.02	+20	+9.96

Table 14. Results of watermark shortcut with ImageNet-V2.

E.3. More Qualitative Examples of Watermark Shortcut

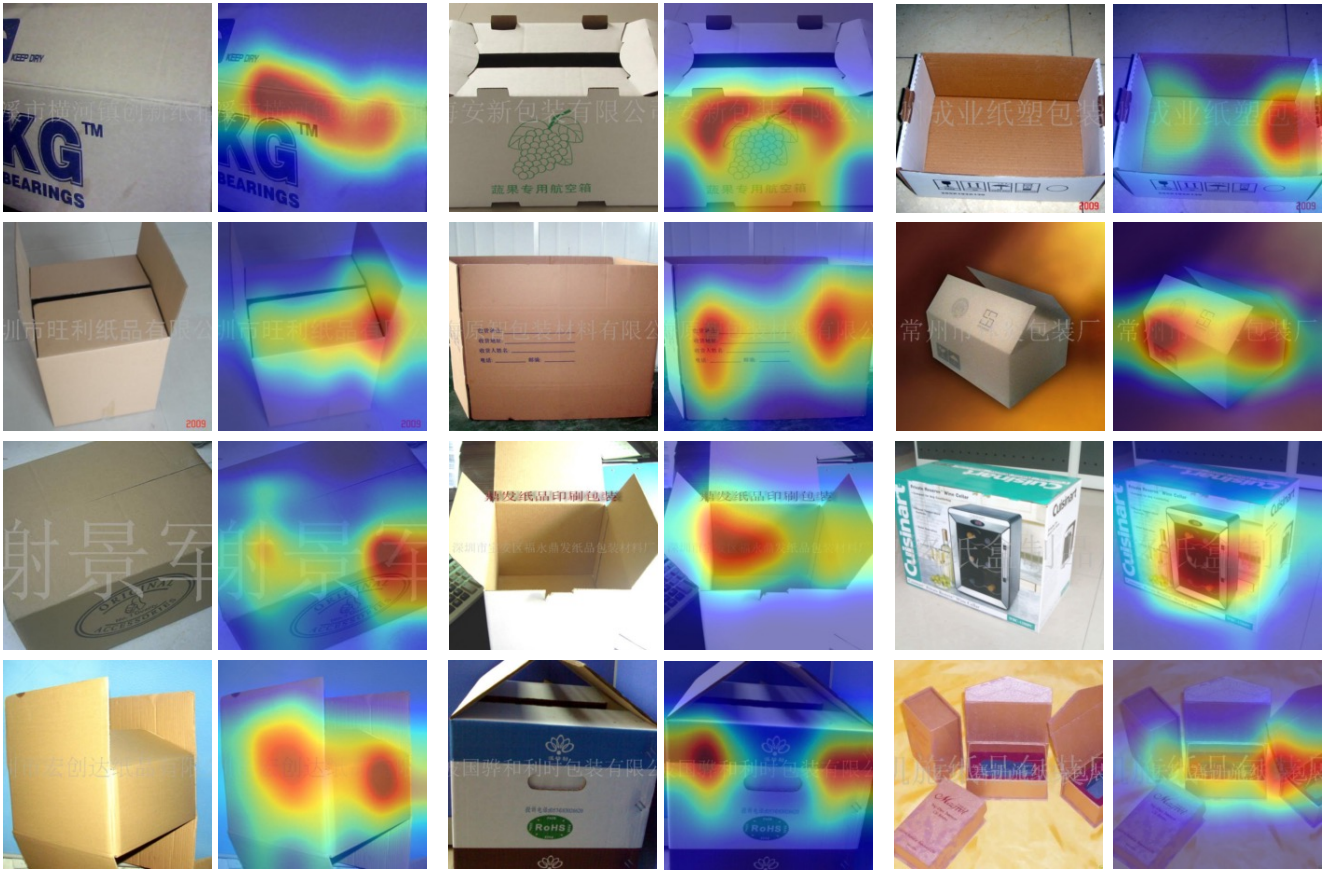
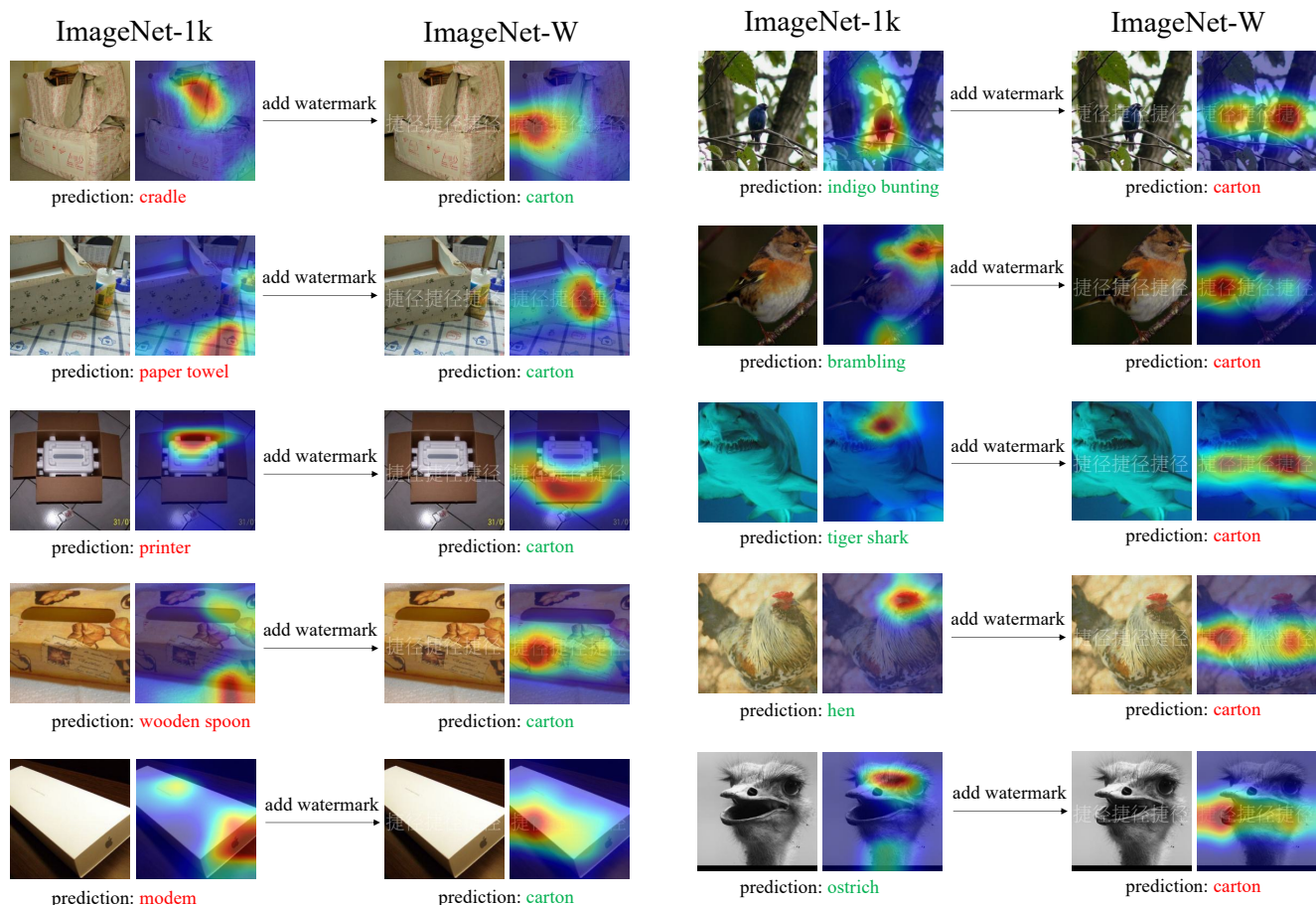


Figure 7. More examples of carton class images with watermark in ImageNet-1k training set. The saliency maps show that ResNet-50 relies on the watermark shortcut to predict carton.

Many Carton Class Images in ImageNet-1k Training Set Contain Watermark We show more watermark examples of carton class images in ImageNet-1k training set. As shown in Fig. 7, these images contain the watermark written in Chinese characters. We also show ResNet-50’s saliency maps [78] for predicting the carton class. While they highlight the watermark region, it may still be hard to interpret because the watermark and the carton object share similar spatial locations. This could be one of the reasons why previous works did not discover this shortcut.

Adding Watermark to Carton Class Images in IN-1k Validation Set (i.e., IN-W) Leads to Carton Class Predictions Our ImageNet-W can better address the difficulty of interpreting the watermark shortcut by providing the counterfactual explanations. In Fig. 8a, we first show carton class images in ImageNet-1k validation set that are predicted incorrectly by ResNet-50 (e.g., cradle, paper towel, etc.). By adding the watermark to the images, we show that not only are the predictions altered to carton but also the highlighted regions of the saliency maps are shifted to the watermark.

Adding Watermark to Non-Carton Class Images in IN-1k Validation Set (i.e., IN-W) Leads to Carton Class Predictions Similarly, we also show the qualitative results for non-carton class images in Fig. 8b. While ResNet-50 makes correct predictions for non-carton class images (e.g., indigo bunting, brambling, hen, etc.) on IN-1k, the predictions are switched to carton class after adding watermarks to the images. Besides, the saliency maps show that the ResNet-50 shifts its attention from the object to the watermark shortcut.



(a) More examples of carton class images. ResNet-50 mispredicts many of them on ImageNet-1k validation set (left column). On ImageNet-W, after adding watermarks to carton class images from ImageNet-1k, ResNet-50 uses watermark as the shortcut to achieve correct predictions (right column).

(b) More examples of images that are not from the carton class. ResNet-50 predicts many of them correctly on ImageNet-1k's validation set (left column). On ImageNet-W, after adding watermarks to carton class images from ImageNet-1k, ResNet-50 uses the watermark as the shortcut to make incorrect predictions as the carton class (right column).

Figure 8. Adding watermarks alters the prediction and focused region of ResNet-50.



Figure 9. Examples of Style Transfer augmentation [27] (TXT Aug) on carton class images from ImageNet-1k training set. Although the augmentation is designed to mitigate the texture shortcut by increasing the “shape bias,” it unexpectedly preserves or amplifies the shape of the watermark.



Figure 10. Examples of background augmentation (BG Aug) [73,93] on carton class images from ImageNet-1k training set. BG Aug is designed to mitigate the background shortcut. However, it preserves the watermarks, leading models to pivot to the watermark shortcut.

Style Transfer (TXT Aug) Preserves or Amplifies the Shape of Watermark In addition to Fig. 1b, we show more examples of style transfer [27] augmentation for carton class images with watermark in Fig. 9. While the technique was originally targeted at mitigating the texture shortcut by randomizing the texture information to increase the shape bias towards the object, the shape of the watermark shortcut, as shown in Fig. 9, is preserved or even amplified. Watermarks in large font sizes (*cf.* first three images in Fig. 9) are still legible after style transfer. The pattern of watermarks in small font size is still retained or even more salient, *e.g.*, the pattern of the transparent watermarks becomes more salient after style transfer when the background is white. This can explain why style transfer (*i.e.*, TXT Aug) amplifies the watermark shortcut results in Tabs. 4 and 15.

Background Augmentation (BG Aug) Preserves the Watermark Shortcut Besides Fig. 1b, we show more examples of background augmentation (BG Aug) [73,93] preserving the watermark shortcut in Fig. 10. Since the watermark is located over the main object, watermarks are still visible when replacing the background with a random one, which explains why BG Aug amplifies the watermark shortcut in Tab. 4. More recently, RobustViT [12] uses the object mask to regularize the model to focus on the object region in the objective function, aiming to mitigate the background shortcut. Although it does not use masks to modify the input image as BG Aug does, we show that it also amplifies the watermark shortcut in Tab. 15 (*cf.* Appendix F.1), which can be explained by the shared spatial locations between watermark and carton object.

F. More Results of Multi-Shortcut Mitigation on ImageNet

F.1. Benchmark More Existing Approaches

End-to-End Training In Sec. 5.2 and Tab. 4, we benchmark existing methods using last layer re-training [46]. Here we show the results of those methods (*i.e.*, Mixup, Cutout, CutMix, AugMix, SD, Style Transfer, LfF, JTT, EIIL, DebiAN) using end-to-end training in Tab. 15. We show that most of them still exhibit the Whac-A-Mole problem by achieving worse shortcut mitigation results. Although Mixup does not amplify shortcuts, its improvement over ERM is still small.

Big Transfer (BiT) We also show the results of Big Transfer (BiT-M) [49], a foundation model pretrained on ImageNet-21k (*i.e.*, excluding 1k classes of ImageNet-1k from the full ImageNet with 22k classes) using ResNet-50v2 [32] architecture. Tab. 15 shows that BiT-M achieves a larger SIN Gap than ERM and barely mitigates the background shortcut.

RobustViT Mitigates Background Shortcut but Amplifies Other Shortcuts RobustViT [12] is a recent work designed to mitigate the background shortcut by optimizing the relevance map based on the object mask. The results in Tab. 15 show that it mitigates the background shortcut but amplifies the watermark shortcut. Besides, it also achieves a worse SIN Gap result for the texture shortcut.

F.2. Results: LLE Using Other Feature Extractors

We further show the results of models using the large ViT-H architecture in Tab. 16. We observed that there is no clear winner among these methods for achieving the best mitigation results on all shortcuts. Our method (LLE) can improve shortcut

		shortcut reliance					
		IN-1k	Watermark		Texture		Background
			IN-W Gap \uparrow	Carton Gap \downarrow	SIN Gap \uparrow	IN-R Gap \uparrow	IN-9 Gap \uparrow
ERM	ResNet-50	76.13	-26.64	+40	-69.03	-55.96	-5.53
Mixup	ResNet-50	76.11	-12.30	+38	-66.81	-53.03	-5.06
CutMix	ResNet-50	78.58	-19.50	+22	-72.86 ($\times 1.06$)	-58.51 ($\times 1.05$)	-6.25 ($\times 1.13$)
Cutout	ResNet-50	77.06	-16.29	+32	-69.95 ($\times 1.01$)	-57.32 ($\times 1.02$)	-5.90 ($\times 1.07$)
AugMix	ResNet-50	77.53	-16.76	+36	-66.38	-51.83	-6.42 ($\times 1.16$)
SD	ResNet-50	70.19	-16.12	+30	-63.63	-59.32 ($\times 1.06$)	-10.89 ($\times 1.97$)
Style Transfer (Texture)	ResNet-50	60.18	-17.31	+52 ($\times 1.30$)	-4.32	-40.76	-7.81 ($\times 1.41$)
LfF	ResNet-50	70.26	-17.57	+40	-64.34	-56.54 ($\times 1.01$)	-8.10 ($\times 1.46$)
JTT	ResNet-50	75.64	-15.74	+32	-69.04	-55.70	-6.75 ($\times 1.22$)
EiIL	ResNet-50	65.42	-19.71	+42 ($\times 1.05$)	-61.27	-57.43 ($\times 1.03$)	-8.66 ($\times 1.57$)
DebiAN	ResNet-50	74.05	-20.00	+30	-67.54	-56.70 ($\times 1.01$)	-7.29 ($\times 1.32$)
BiT-M (IN-21k)	ResNet-50v2	82.32	-8.63	+28	-73.69 ($\times 1.07$)	-51.19	-5.25
ERM	ViT-B/16	81.07	-6.69	+26	-62.67	-50.36	-5.36
RobustViT (Background)	ViT-B/16	80.33	-7.35 ($\times 1.10$)	+30 ($\times 1.15$)	-64.06 ($\times 1.02$)	-45.64	-5.01

Table 15. More multi-shortcut mitigation results on ImageNet. Note that methods from ERM to DebiAN use end-to-end training, which is different from the last layer re-training setting in Tab. 4. BiT-M is a foundation model pretrained on ImageNet-21k (IN-21k). RobustViT fine-tunes an ERM to mitigate the background shortcut.

			shortcut reliance				
			Watermark		Texture		Background
train data	IN-1k		IN-W Gap	Carton Gap	SIN Gap	IN-R Gap	IN-9 Gap
SWAG (LP)	IG-3.6B	85.74	-4.89	+8	-59.99	-8.80	-7.86
SWAG (FT)	IG-3.6B	88.54	-3.09	+18	-62.22	-9.37	-3.19
CLIP (zero-shot)	LAION-2B	77.90	-3.61	+16	-59.47	-5.61	-3.71
MAE (FT)	IN-1k	86.89	-3.48	+30	-62.29	-33.15	-3.24
MAE+LLE (ours)	IN-1k	86.84	-1.11	+28	-55.69	-30.95	-2.35

Table 16. Multi-shortcut mitigation results on ImageNet with ViT-H network architecture. LP and FT stand for linear probing and fine-tuning on ImageNet-1k, respectively. Note that there is no ERM (supervised training) available with ViT-H on ImageNet-1k.

mitigation results over MAE in all metrics. Our method can even beat methods using extra pretraining data (*i.e.*, SWAG and CLIP) in IN-W Gap, SIN Gap, and IN-9 Gap.

Besides, we also show the results of LLE using SWAG (FT) in ViT-B/16 architecture in Tab. 17. While SWAG (LP) and SWAG (FT) suffer the Whac-A-Mole dilemma, LLE consistently mitigates multiple shortcuts jointly over ERM and SWAG (FT). Besides, we also show SWAG (FT) + LLE with edge augmentation (Edge Aug) and the results on ImageNet-Sketch. More details are introduced below (*cf.* Appendix F.3).

F.3. Results of LLE on ImageNet-Sketch

Results: ImageNet-Sketch We further show the results of LLE on ImageNet-Sketch [87] (IN-Sketch), another OOD variant of ImageNet containing sketch images in 1000 ImageNet classes. We use IN-Sketch Gap, the accuracy drop from IN-1k to IN-Sketch, to measure mitigation of color and texture shortcuts. The results in Tab. 17 show that our LLE method consistently improves the results over ERM, MAE, and SWAG (FT).

Edge Augmentation While style transfer augmentation could be suboptimal for mitigating the color and texture shortcuts measured by IN-Sketch, we propose edge augmentation (Edge Aug) to improve the results further. Concretely, we use [66] to detect edges on images from ImageNet-1k training set. The examples are shown in Fig. 11, where we observe that color and texture information is successfully removed via edge detection. Similar to style transfer and background augmentation (*cf.* Fig. 1b), we still observe the amplified or preserved saliency of the watermark (*cf.* carton class image in Fig. 11). The edge augmentation is used to train an additional last layer in the classifier ensemble. The results in Tab. 17 show that using

			shortcut reliance					
	train data	IN-1k	Watermark		Texture		Background	Color and Texture
			IN-W Gap	Carton Gap	SIN Gap	IN-R Gap	IN-9 Gap	IN-Sketch Gap
<i>arch: ResNet-50</i>								
ERM	IN-1k	76.39	-25.40	+30	-69.43	-56.22	-5.19	-52.32
LLE (ours)	IN-1k	76.25	-6.18	+10	-61.02	-54.89	-3.82	-51.56
LLE (ours) + Edge Aug	IN-1k	76.24	-6.18	+10	-61.52	-53.69	-3.95	-48.25
<i>arch: ViT-B/16</i>								
ERM	IN-1k	81.07	-6.69	+26	-62.60	-50.36	-5.36	-51.67
SWAG (LP)	IG-3.6B	81.89	-7.76 ($\times 1.16$)	+18	-67.33 ($\times 1.08$)	-19.79	-10.39 ($\times 1.94$)	-32.22
SWAG (FT)	IG-3.6B	85.29	-5.43	+24	-66.99 ($\times 1.07$)	-29.55	-4.44	-42.58
SWAG (FT) + LLE (ours)	IG-3.6B	85.37	-2.50	+8	-60.92	-28.37	-3.19	-41.52
SWAG (FT) + LLE (ours) + Edge Aug	IG-3.6B	85.31	-2.48	+12	-61.24	-27.78	-3.28	-38.37
MAE (FT)	IN-1k	83.72	-4.60	+24	-65.20 ($\times 1.04$)	-47.10	-4.45	-47.77
MAE + LLE (ours)	IN-1k	83.68	-2.48	+6	-58.78	-44.96	-3.70	-46.70
MAE + LLE (ours) + Edge Aug	IN-1k	83.69	-2.54	+6	-59.04	-43.97	-3.70	-43.17
<i>arch: ViT-L/16</i>								
ERM	IN-1k	79.65	-6.14	+34	-61.43	-53.17	-6.50	-52.40
MAE (FT)	IN-1k	85.95	-4.36	+22	-62.48 ($\times 1.02$)	-36.46	-3.53	-40.29
MAE + LLE (ours)	IN-1k	85.84	-1.74	+12	-56.32	-34.64	-2.77	-39.14
MAE + LLE (ours) + Edge Aug	IN-1k	85.84	-1.76	+16	-56.52	-33.76	-2.94	-36.45

Table 17. Ablation study of adding edge augmentation (Edge Aug) to LLE. Edge Aug further improves the results on ImageNet-Sketch.

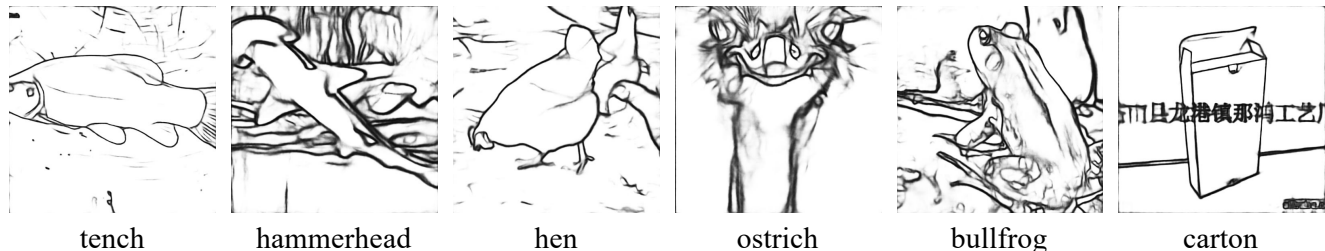


Figure 11. Example images of edge augmentation for ImageNet-1k training set to mitigate color and texture shortcuts. The ground-truth class name is shown below each image.

Edge Aug can further close the In-Sketch Gap and IN-R Gap—IN-R also contains sketch images. The results demonstrate the effectiveness of designing targeted augmentation to tackle the known type of shortcut.

F.4. Top-1 Accuracy of LLE on OOD Variant of ImageNet

In this work, we mainly use the gap of accuracy between IN-1k to OOD variants of ImageNet as the metric. We also show the results of LLE in top-1 accuracy on OOD variants of ImageNet in Tab. 18, which can help future research to compare with LLE in top-1 accuracy.

Note that we do not include the top-1 accuracy on ImageNet-W. Although existing models suffer a performance drop from IN-1k to IN-W, an increased IN-W accuracy over IN-1k, which future works may achieve, also indicates the watermark shortcut reliance. Because of the counterfactual nature between IN-1k and IN-W, we encourage future works to use IN-W Gap and Carton Gap to report the watermark shortcut mitigation results, where closer to zero gaps indicate better results.

F.5. Results of LLE on Other OOD Variants of ImageNet

We also show the results of LLE on other OOD variants of ImageNet, including ImageNet-A [37] (IN-A), ImageNetV2 [69] (IN-V2), ObjectNet [9], and ImageNet-D [71,72] (IN-D). IN-D has rendition images similar to IN-R except for having additional domain annotations, *e.g.*, clipart, infograph, *etc.* Besides, IN-D also has real-domain images (*i.e.*, IN-D real). We report the top-1 accuracy on IN-A, IN-V2, ObjectNet, and IN-D clipart to IN-D sketch. Regarding the types of shortcut reliance, ObjectNet measures the robustness against unusual background, viewpoint, and rotation. The results from IN-D clipart to IN-D sketch measure the robustness against the texture shortcut. The remaining results, *i.e.*, IN-A, IN-V2, IN-D real, do not explicitly measure the robustness against specific shortcuts. Therefore, we denote their shortcut reliance type as “unknown.”

	arch	train data	IN-1k	shortcut reliance			
				Texture		Background	Color and Texture
				SIN	IN-R	Mixed-Rand	IN-Sketch
LLE	ResNet-50	IN-1k	76.25	15.25	37.31	84.40	24.67
LLE + Edge Aug	ResNet-50	IN-1k	76.24	14.72	38.43	84.30	27.99
SWAG (FT) + LLE	ViT-B/16	IG-3.6B	85.37	24.45	68.14	90.12	43.85
SWAG (FT) + LLE + Edge Aug	ViT-B/16	IG-3.6B	85.31	24.07	68.70	89.98	46.94
MAE + LLE	ViT-B/16	IN-1k	83.68	24.90	50.84	89.41	36.98
MAE + LLE + Edge Aug	ViT-B/16	IN-1k	83.69	24.65	51.85	89.36	40.52
MAE + LLE	ViT-L/16	IN-1k	85.84	29.52	62.24	91.58	46.70
MAE + LLE + Edge Aug	ViT-L/16	IN-1k	85.84	29.32	63.13	91.41	49.39
MAE + LLE	ViT-H/14	IN-1k	86.84	31.15	66.21	93.01	50.60
MAE + LLE + Edge Aug	ViT-H/14	IN-1k	86.84	30.94	66.89	92.86	53.39

Table 18. Top-1 accuracy results of Last Layer Ensemble (LLE) on OOD variants of ImageNet.

	arch	(pre)training data	shortcut reliance											
			unknown		background, viewpoint, rotation			texture			unknown	IN-D (mDE) ↓		
			IN-A	IN-V2	ObjectNet		IN-D clipart	IN-D infograph	IN-D painting	IN-D quickdraw	IN-D sketch		IN-D real	
ERM	ResNet-50	IN-1k	0.02	63.48	36.10			23.94	10.69	34.83	0.83	17.77	59.86	88.27
LLE	ResNet-50	IN-1k	0.12	63.34	36.67			25.86	11.35	36.86	0.85	19.57	60.60	86.79
LLE + Edge Aug	ResNet-50	IN-1k	0.09	63.05	36.67			26.31	11.29	36.82	0.92	20.72	60.57	86.50
ERM	ViT-B/16	IN-1k	20.88	69.56	39.89			29.87	13.62	41.37	1.13	21.86	62.75	83.53
SWAG (FT)	ViT-B/16	IG-3.6B	53.01	75.58	53.90			49.54	20.09	52.88	2.53	39.34	68.17	70.99
SWAG (FT) + LLE	ViT-B/16	IG-3.6B	53.71	75.75	54.48			51.18	21.63	54.88	3.19	41.09	69.12	69.25
SWAG (FT) + LLE + Edge Aug	ViT-B/16	IG-3.6B	53.75	75.68	54.55			51.69	21.43	54.93	3.59	41.95	69.20	68.93
MAE (FT)	ViT-B/16	IN-1k	35.81	73.20	47.30			34.11	15.27	44.30	1.17	27.14	64.92	80.15
MAE (FT) + LLE	ViT-B/16	IN-1k	36.88	73.06	47.63			35.25	16.37	45.90	1.25	28.66	65.47	78.93
MAE (FT) + LLE + Edge Aug	ViT-B/16	IN-1k	37.00	72.94	47.79			35.73	16.10	45.97	1.34	29.65	65.52	78.66
ERM	ViT-L/16	IN-1k	16.64	67.49	36.79			27.68	12.45	39.47	0.58	19.40	62.04	85.32
MAE (FT)	ViT-L/16	IN-1k	57.07	76.65	55.31			42.64	18.05	50.14	3.12	36.87	66.66	74.10
MAE (FT) + LLE	ViT-L/16	IN-1k	56.65	76.74	55.46			43.95	19.31	51.67	3.27	38.05	67.29	72.87
MAE (FT) + LLE + Edge Aug	ViT-L/16	IN-1k	56.77	76.66	55.65			44.24	19.06	51.81	3.44	38.88	67.29	72.65
MAE (FT)	ViT-H/14	IN-1k	68.17	78.46	60.47			43.69	19.10	51.29	3.89	39.17	67.61	72.63
MAE (FT) + LLE	ViT-H/14	IN-1k	68.27	78.34	60.61			45.40	20.80	52.94	4.24	40.75	68.20	71.12
MAE (FT) + LLE + Edge Aug	ViT-H/14	IN-1k	68.35	78.32	60.78			45.76	20.66	53.06	4.40	41.60	68.23	70.86

Table 19. Results of LLE on other OOD variants of ImageNet, *i.e.*, ImageNet-A (IN-A), ImageNetV2 (IN-V2), ObjectNet, and ImageNet-D (IN-D). Except for IN-D overall results (*i.e.*, last column), all other results are in top-1 accuracy. The overall IN-D results are reported in mDE, where lower numbers indicate better results (↓).

The results are shown in Tab. 19. On both ObjectNet and IN-D datasets, LLE consistently improves the results over various baselines (*i.e.*, ERM, SWAG (FT), and MAE (FT)) in different network architectures. When the shortcut type is unknown, LLE achieves comparable results against the baselines with slight performance improvement or drop depending on the architectures and pretraining datasets. Note that LLE is designed for mitigating multiple *known* shortcuts (*cf.* Sec. 4). Therefore, it may not improve the results when the types of shortcuts remain unknown. However, due to the theoretical impossibility of inferring shortcut labels [58] and the practical difficulty of mitigating multiple unknown shortcuts, we encourage future research to tackle this problem by first interpreting the distributional shift on IN-A or IN-V2 before performing mitigation (more discussion in Appendix H).

G. CutMix Amplifies Background Shortcut

Results of CutMix on Waterbirds On UrbanCars (*cf.* Tab. 5) and ImageNet (*cf.* Tabs. 4 and 15), we observe that CutMix [94] amplifies the background shortcut. We further show its background shortcut reliance on Waterbirds dataset. We use the following metrics on Waterbirds: (1) Average Group Accuracy: the unweighted average results over four groups ($\{\text{waterbird, landbird}\} \times \{\text{water background, land background}\}$); (2) Worst Group Accuracy: the lowest per group accuracy result. For this experiment on Waterbirds, we use the experiment setting on UrbanCars (*cf.* Appendix B.3). Tab. 20 shows that CutMix achieves worse results of mitigating the background shortcut than ERM. Other techniques, *i.e.*, Mixup and Cutout, slightly mitigates background shortcut on Waterbirds.

	Average Group Accuracy (%)	Worst Group Accuracy (%)
ERM	87.19	73.88
Mixup ($\alpha = 0.05$)	87.76	75.73
Cutout ($p = 0.1$)	88.57	74.87
CutMix ($\alpha = 1.0$)	74.51 (-12.68)	47.38 (-26.50)

Table 20. Results of standard augmentation and regularization on Waterbirds [74] dataset. CutMix amplifies the background shortcut on the Waterbirds dataset. (·): hyperparameter used in each approach.

Explaining the Background Shortcut Reliance of CutMix Since CutMix consistently amplifies the background shortcut on three datasets (*i.e.*, UrbanCars, Waterbirds, and ImageNet), we take a closer look at its augmentation and regularization strategy. In terms of augmentation, CutMix crops a rectangular patch from one image and pastes it to the other to create the augmented image. In the regularization, the ground-truth label for the augmented image is the linear interpolation of ground-truth labels of two source images, where the interpolation co-efficient (*i.e.*, called combination ratio λ in CutMix) is proportional to the area of the patch. In this way, the network is regularized to predict the probability over classes that is proportional to the area in the image. Therefore, when the background takes the larger area in the image, the model predicts more on the background class instead of the smaller foreground object, leading to an amplified background shortcut reliance.

H. Discussion

H.1. End-to-End Training vs. Last Layer Re-Training—A Multi-Shortcut Mitigation Perspective

Most existing shortcut mitigation methods (*e.g.*, gDRO [74], SUBG [39], DI [89], JTT [59], EIL [15], LfF [61], and DebiAN [54]) train the model end-to-end. Recently, Kirichenko *et al.* [46] propose Deep Feature Reweighting (DFR), which only retrains the last classification layer of the ERM model, *i.e.*, the feature extractor of the ERM model is frozen. DFR enjoys the advantage of efficient training compared to traditional end-to-end training approaches, which motivates us to propose our Last Layer Ensemble (LLE) method to mitigate multiple shortcuts efficiently.

However, one may worry that methods based on last layer re-training may achieve suboptimal shortcut mitigation results compared to end-to-end training approaches because the former’s performance is decided by (1) how much the intended features can be extracted by the feature extractor and (2) whether the feature extractor can disentangle the intended and shortcut features. Empirically, DFR still has some gaps in combating distributional shift compared to end-to-end training methods (*e.g.*, results of ImageNet-R and ImageNet-C in Table 3 of [46]).

While Kirichenko *et al.* [46] compare the two training strategies in the single-shortcut setting, our work provides a new multi-shortcut mitigation perspective on this problem. Concretely, we compare the results of two methods—SUBG [39] (*i.e.*, an end-to-end training method) and DFR [46] (*i.e.*, a last layer re-training method) because DFR retrains the last classification layer with SUBG method. In other words, the only difference between SUBG and DFR is the training strategy, making an apples-to-apples comparison. The results of two methods on UrbanCars in Tab. 5 reveal an interesting finding. When labels of both shortcuts are used, SUBG outperforms DFR in mitigating both shortcuts. However, if labels of either shortcut are not used, SUBG amplifies the unlabeled shortcut much more significantly compared to DFR.

Therefore, from the multi-shortcut mitigation perspective, we find **last layer re-training is a more “conservative” strategy—although the results of mitigating the labeled shortcuts may not be optimal, it has a lower risk of significantly amplifying the unlabeled shortcuts**, which is more typical in in-the-wild datasets where types and numbers of shortcuts usually remain unknown.

H.2. Can the problem of the watermark shortcut be addressed through data cleaning?

We believe that using data cleaning to address the watermark shortcut problem is suboptimal for three reasons. First, it is infeasible to remove watermark images without watermark labels. Using watermark detection models may have problems because they may have shortcuts in themselves, *e.g.*, working well for English but not Chinese watermarks. Second, removing watermarks from images (*e.g.*, using in-painting) requires masks, which is non-trivial. Finally, removing watermark images shrinks the training set size and may amplify geographical biases. For example, we find that images with Chinese watermarks mainly from online shopping websites in China. Simply discarding these images could create performance disparity across different geographical regions [16,70].

H.3. Recommendation and Future Direction

To future shortcut mitigation practitioners, we recommend the community drop the unrealistic single-shortcut assumption and be aware of the multiple-shortcut problem by having a sanity check on various inductive biases in model design, such as the usage of shortcut labels, assumption of shortcut learning during training, data augmentation, regularization, *etc.*

For future shortcut mitigation dataset creators, a broader range of factors of variations (FoV) needs to be studied since some FoVs could serve as multiple shortcuts learned by models. This can be achieved by (1) manually choosing various FoVs under the controlled setting [9,23,38,40,52,77] or (2) developing better approaches to detect and interpret shortcuts [2,6,19,24,43,55,83] on in-the-wild datasets.

Although our work mainly focuses on the shortcut mitigation task, the importance and challenge of multiple shortcuts also apply to the shortcut detection task. For example, Eyuboglu *et al.* [24] design a shortcut detection benchmark based on CelebA, where only a single shortcut exists. Specifically, they achieve this by amplifying the correlation strength of the spurious correlation between the target attribute and the shortcut attribute. Therefore, whether or not existing shortcut detection approaches can detect multiple shortcuts is underexplored and is a promising future direction.

H.4. Limitations

Admittedly, our work has limitations. For example, Last Layer Ensemble (LLE) does not address the problem of unknown types of shortcuts, which LLE may amplify. However, since mitigating unknown types of shortcuts without any inductive biases is still a theoretical [58] and practical challenge, we advocate a human-in-the-loop solution. That is, detecting and interpreting shortcuts at the first stage. Then, LLE can be applied to mitigate the detected shortcuts.