[Re] Don't Judge an Object by Its Context: Learning to Overcome Contextual Bias

Anonymous Author(s)
Affiliation
Address
email

Reproducibility Summary

Scope of Reproducibility

Singh et al. [9] point out the dangers of *contextual bias* in visual recognition datasets. They propose two methods, *CAM-based* and *feature-split*, that better recognize an object or attribute in the absence of its typical context while maintaining competitive within-context accuracy. To verify their performance, we attempted to reproduce all 12 tables in the original paper, including those in the appendix. We also conducted additional experiments to better understand the proposed methods, including increasing the regularization in *CAM-based* and removing the weighted loss in *feature-split*.

Methodology

As the original code was not made available, we implemented the entire pipeline from scratch in PyTorch 1.7.0. Our implementation is based on the paper and email exchanges with the authors.[1] We spent approximately four months reproducing the paper, with the first two months focused on implementing all 10 methods studied in the paper and the next two months focused on reproducing the experiments in the paper and refining our implementation. Total training times for each method ranged from 35–43 hours on COCO-Stuff [1], 22–29 hours on DeepFashion [7], and 7–8 hours on Animals with Attributes [10] on a single RTX 3090 GPU.

Results

We found that both proposed methods in the original paper help mitigate contextual bias, although for some methods, we could not completely replicate the quantitative results in the paper even after completing an extensive hyperparameter search. For example, on COCO-Stuff, DeepFashion, and UnRel, our *feature-split* model achieved an increase in accuracy on out-of-context images over the standard baseline, whereas on AwA, we saw a drop in performance. For the proposed *CAM-based* method, we were able to reproduce the original paper's results to within 0.5% mAP.

What was easy

Overall, it was easy to follow the explanation and reasoning of the experiments. The implementation of most (7 of 10) methods was straightforward, especially after we received additional details from the original authors.

What was difficult

Since there was no existing code, we spent considerable time and effort re-implementing the entire pipeline from scratch and making sure that most, if not all, training/evaluation details are true to the experiments in the paper. For several methods, we went through many iterations of experiments until we were confident that our implementation was accurate.

Communication with original authors

We reached out to the authors several times via email to ask for clarifications and additional implementation details. The authors were very responsive to our questions, and we are extremely grateful for their detailed and timely responses.

[1] Please find our code in the supplementary material; we will add the link to our public code repository after the review process.

Submitted to ML Reproducibility Challenge 2020. Do not distribute.

1 Introduction

Most prominent vision datasets are afflicted by *contextual bias*. For example, "microwave" typically is found in kitchens, which also contain objects like "refrigerator" and "oven." Such co-occurrence patterns may inadvertently induce contextual bias in datasets, which could consequently seep into models trained on them. When models overly rely on context, they may not generalize to settings where typical co-occurrence patterns are absent. The original paper by Singh et al. [9] proposes two methods for mitigating such contextual biases and improving the robustness of the learnt feature representations. The paper demonstrates their methods on multi-label object and attribute classification tasks, using the COCO-Stuff [1], DeepFashion [7], Animals with Attributes (AwA) [10], and UnRel [8] datasets. Our exploration centers on four main directions:

First, we trained the baseline classifier presented in the paper (Section 2.1 for implementation; Sections 2.3-2.4 for results). Due to likely implementation discrepancies, our results differed from the original paper by 0.6–3.1% mAP on COCO-Stuff, by 0.7–1.4% top-3 recall on DeepFashion, and by 0.1–3.2% mAP on AwA (Table 2). We ran a hyperparameter search (Appendix C), which yielded a significant (1.4–3.6%) improvement on DeepFashion.

Next, we identified the *biased categories* in each dataset, i.e., visual categories that suffer from contextual bias. We followed the proposed method of using the baseline classifier to identify these categories, and discovered that the classifier implementation has a non-trivial effect. For COCO-Stuff, 18 of the top-20 categories we identified matched the original paper's top-20 categories (10 on DeepFashion, 18 on AwA; Section 2.2). Nevertheless, the categories we identified appear reasonable (e.g., "fork" co-occurs with "dining table"; Appendix B). As training and evaluation of most methods depend on the biased categories, we used the paper's biased categories for subsequent experiments.

Third, we checked the main claim of the paper, that the proposed *CAM-based* and *feature-split* methods help improve recognition of biased categories in the absence of their context (Section 3). On COCO-Stuff, DeepFashion, and UnRel, we were able to reproduce the improvements gained from the proposed *feature-split* method towards reducing contextual bias, whereas on AwA, we saw a drop in performance. The proposed *CAM-based* method, which was only applied to COCO-Stuff, also helped reduce contextual bias, though not as significantly as the *feature-split* method. For this method, we reproduced the original paper's results to within 0.5% mAP (Section 3.5). We also successfully reproduced the paper's weight similarity analysis, as well as the qualitative analyses on class activation maps (CAMs) [12].

Lastly, we ran additional experiments and ablation studies (Section 3.6). These revealed that the regularization term in the *CAM-based* method and the weighted loss in the *feature-split* method are central to the methods' performance. We also observed that varying the feature subspace size influences the *feature-split* method accuracy.

2 Reproducing the *standard* baseline and the biased category pairs

The first step in reproducing the original paper is doing "stage 1" training. This stage involves training a *standard* multi-label classifier with the binary cross entropy loss on the COCO-Stuff, DeepFashion, and AwA datasets. We describe how we obtained and processed the datasets in Section A of the appendix. The *standard* model is used to identify the biased categories and serves as a starting point for all "stage 2" methods, i.e., the proposed *CAM-based* and *feature-split* methods and 7 other strong baselines introduced in Section 3.

2.1 Implementation and training details

According to the original paper, all models use ResNet-50 [4] pre-trained on ImageNet [3] as a backbone and are optimized with stochastic gradient descent (SGD) and a batch size of 200. Each *standard* model is optimized with an initial learning rate of 0.1, later dropped to 0.01 following a standard step decay process. The input images are randomly resize-cropped to 224×224 and randomly flipped horizontally during training. We also received additional details from the authors that SGD is used with a momentum of 0.9 and no weight decay. The COCO-Stuff *standard* model is trained for 100 epochs with the learning rate reduced from 0.1 to 0.01 after epoch 60. The DeepFashion *standard* model is trained for 50 epochs with the learning rate reduced after epoch 30. The AwA *standard* model is trained for 20 epochs with the learning rate reduced after epoch 10.

After training with the paper's hyperparameters, we found that our reproduced *standard* models for COCO-Stuff and AwA were consistently underperforming against the results in the paper. Thus, we also tried varying the learning rate, weight decay, and the epoch at which the learning rate is dropped to achieve the best possible results. Further details can be found in Section C of the appendix. On both COCO-Stuff and AwA, our hyperparameter search ended up reconfirming the original paper's hyperparameters as the optimal ones; for DeepFashion, we were able to find an improvement. The original, reproduced and tuned results are shown in Table 1, following explanations of biased categories identification (Section 2.2) and evaluation details (Section 2.3).

2.2 Biased categories identification

The paper identifies the top-20 (b, c) pairs of biased categories for each dataset, where b is the category suffering from contextual bias and c is the associated context category. This identification is crucial as it concretely defines the contextual bias the paper aims to tackle, and influences the training of the "stage 2" models and evaluation of all models.

The paper defines *bias* between two categories b and z as the ratio between average prediction probabilities of b when it occurs with and without z. Note that this definition of bias requires a trained model, unlike the more common definition of bias that only requires co-occurrence counts in a dataset [11]. Following the paper description, we used a *standard* model trained on an 80-20 split for COCO-Stuff and one trained on the full training set for DeepFashion and AwA. For each category b in a given dataset, we calculated the bias between b and its frequently co-occuring categories, and defined category c as the context category that most biases b, i.e. has the highest bias value. Bias is calculated on the 20 split for COCO-Stuff, the validation set for DeepFashion, and the test set for AwA.[2] After the bias calculation, we identified 20 (b, c) pairs with the highest bias values. The paper emphasizes that this definition of bias is directional; it only captures the bias c incurs on b and not the other way around.

We compared our pairs to the paper's in Tables A1, A2, A3 in the appendix. Out of 20 biased categories, 2 of ours differed from the paper's for COCO-Stuff, 10 differed for DeepFashion, and 2 differed for AwA. The variability is expected, as bias is defined as a ratio of a trained model's average prediction probabilities which will vary across different models. Nonetheless, we found our pairs to also be reasonable, as our biased categories occur frequently with their context categories and rarely without them. See Section B in the appendix for details.

2.3 Evaluation details

The paper does not specify image preprocessing or model selection. Following common practice, we resize an image so that its smaller edge is 256 and then apply one of two 224x224 cropping methods: a center-crop or a ten-crop. Both are deterministic procedures. We observed that results with center-crop are consistently better and closer to the paper's results, hence for all experiments, we report results using center-crop. In our email communications, the authors also specified that they use the model at the end of training as the final model. We confirmed that this is a reasonable model selection method after trying three other selection methods, described in Section D of the appendix.

We emphasize that model evaluation is dependent on the identified biased category pairs. For each (b, c) pair, the test set can be divided into three sets: *co-occurring* images that contain both b and c, *exclusive* images that contain b but not c, and *other* images that do not contain b. Then for each (b, c) pair, the paper constructs two test distributions: 1) the "exclusive" distribution containing *exclusive* and *other* images and 2) the "co-occur" distribution containing *co-occurring* and *other* images. We suspect that *other* images are included in both distributions because otherwise, both distributions would have small sizes and only consist of positive images where b occurs, disabling the mAP calculation. The test distribution sizes can be calculated from the co-occurring and exclusive image counts in Tables A1, A2, A3 in the appendix. As an example, for the (ski, person) pair in COCO-Stuff, there are 984 co-occuring, 9 exclusive, and 39,511 other images in the test set. Hence, there are $9 + 39,511 = 39,520$ images in the "exclusive" distribution and $984 + 39,511 = 40,495$ images in the "co-occur" distribution. For COCO-Stuff, we also report results on the entire test set (40,504 images) for 60 non-biased object categories and for all 171 categories, following the paper.

[2] We received additional information from the original authors that they restricted their COCO-Stuff biased categories to 80 object/thing categories and performed manual cleaning of the DeepFashion biased category pairs.

Dataset (Metric)	Model	Exclusive		Co-occur		Non-biased		All
		Paper BC	Our BC	Paper BC	Our BC	Paper BC	Our BC	
COCO-Stuff (mAP)	Paper	**24.5**	-	**66.2**	-	**75.4**	-	**57.2**
	Ours (paper params*)	23.9	20.6	65.0	63.7	72.3	72.9	55.7
DeepFashion (top-3 recall)	Paper	4.9	-	17.8	-	-	-	-
	Ours (paper params)	5.6	5.0	19.2	15.0	-	-	-
	Ours (tuned params)	**7.0**	6.3	**22.8**	18.4	-	-	-
AwA (mAP)	Paper	19.4	-	**72.2**	-	-	-	-
	Ours (paper params*)	**19.5**	21.7	69.0	69.9	-	-	-

Table 1: Reproduced *standard* baseline results on three datasets. We evaluate the models on different subsets of categories/images ("exclusive" and "co-occur" distributions for the 20 biased categories and the entire test set for non-biased and all categories; Section 2.3), both using the paper's and our identified biased category (BC) pairs. *On COCO-Stuff and AwA, hyperparameter tuning did not improve on the original paper's hyperparameters.

For COCO-Stuff and AwA, we calculate the average precision (AP) for each biased category b, and report the mean AP (mAP) for each test distribution. For DeepFashion, we calculate the per-category top-3 recall and report the mean value for each test distribution. Higher values indicate a better classifier for both metrics.

2.4 Results

In Table 1, we report the original, reproduced, and tuned results with the paper's and our 20 most biased category pairs. Evaluated on the paper's pairs, our best COCO-Stuff model underperforms the paper's by 1-3%, our best DeepFashion model outperforms by 2-5%, and our best AwA underperforms on the "co-occur" distribution by 3.2% and matches the "exclusive" distribution within 0.1%. When we evaluate the same models on our biased category pairs, we get similar results for the AwA model, slightly worse results for the DeepFashion model, and significantly worse results for the COCO-Stuff model. Due to this big drop in performance for COCO-Stuff, which we suspect is caused by the discrepancy in the identified biased category pairs, we choose to use the paper's pairs for training and evaluation in the subsequent sections. Overall, we conclude that the paper's *standard* baseline results are reproducible as we were able to train models within a reasonable margin of error.

3 Reproducing the "stage 2" methods: CAM-based, feature-split, and strong baselines

In this section, we describe our efforts in reproducing methods that aim to mitigate contextual bias: namely, the *CAM-based* and *feature-split* methods proposed by the original authors and 7 other strong baselines. These are referred to as "stage 2" methods because they are trained on top of the "stage 1" *standard* model (except for one strong baseline). Apart from the *feature-split* method, which we discussed with the authors, all other implementations were based entirely on our interpretation of their descriptions in the original paper.

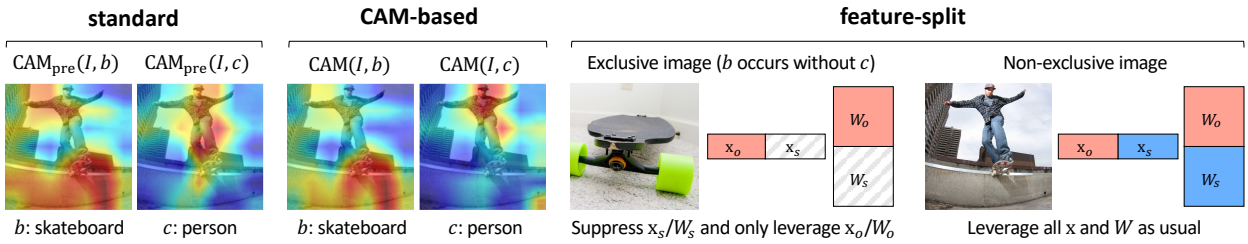


Figure 1: Overview of the proposed methods. The *CAM-based* methods enforces a minimal overlap between the (b, c) CAMs, while preventing them from drifting too far from $\mathrm{CAM}_{\mathrm{pre}}$ (CAMs of the *standard* model). The *feature-split* method suppresses context for exclusive images by disabling backpropagation through W_s and setting x_s to a constant value; for non-exclusive images, it uses everything as usual.

3.1 The first proposed *CAM-based* method

The *CAM-based* method operates on the following premise: as b almost always co-occurs with c, the network may learn to inadvertently rely on pixels corresponding to c to predict b. The paper hypothesizes that one way to overcome this issue is to explicitly force the network to rely less on c's pixel regions. This method uses class activation maps (CAMs) [12] as a proxy for object localization information. For an image I and category r, $\mathrm{CAM}(I, r)$ indicates the discriminative image regions used by a deep network to identify r. For each biased category pair (b, c), a minimal overlap of their CAMs is enforced via the loss term:

$$L_O = \sum_{I \in \mathbb{I}_b \cap \mathbb{I}_c} \mathrm{CAM}(I,b) \odot \mathrm{CAM}(I,c), \tag{1}$$

where $\odot$ denotes element-wise multiplication and $\mathbb{I}_b \cap \mathbb{I}_c$ is a set of images where both b and c appear. To prevent a trivial solution where the CAMs of b and c drift apart from the actual pixel regions, the paper uses a regularization term to keep the category's CAMs close to $\mathrm{CAM}_{\mathrm{pre}}$, produced using a separate network trained offline:

$$L_R = \sum_{I \in \mathbb{I}_b \cap \mathbb{I}_c} |\mathrm{CAM}_{\mathrm{pre}}(I,b) - \mathrm{CAM}(I,b)| + |\mathrm{CAM}_{\mathrm{pre}}(I,c) - \mathrm{CAM}(I,c)|. \tag{2}$$

In our implementation, we separate a batch into two small batches during training, one with and one without co-occurrences. A sample is put into the *co-occurrence* batch if any of the 20 biased categories co-occurs with its context. For the co-occurrence batch, we compute CAM with the current model being trained and $\mathrm{CAM}_{\mathrm{pre}}$ with the trained *standard* model, using the official CAM implementation: `https://github.com/zhoubolei/CAM`. We update the model parameters with the following loss, where L_{BCE} is the binary cross entropy loss:

$$L_{\mathrm{CAM}} = \lambda_1 L_O + \lambda_2 L_R + L_{\mathrm{BCE}}. \tag{3}$$

For the *other* batch without any co-occurrences, we update the model parameters with L_{BCE}. With the hyperparameters reported in the paper, $\lambda_1 = 0.1$ and $\lambda_2 = 0.01$, we got underwhelming results and degenerate CAMs that drifted far from the actual pixel regions. Hence, we tried increasing the regularization weight λ_2 (0.01, 0.05, 0.1, 0.5, 1.0, 5.0) and achieved the best results with $\lambda_2 = 0.1$, which are reported in Table 2.

3.2 The second proposed *feature-split* method

By discouraging mutual spatial overlap, the *CAM-based* approach may not be able to leverage useful information from the pixel regions surrounding the context. Thus, the paper proposes a second method that splits the feature space into two subspaces to separately represent category and context, while posing no constraints on their spatial extents. Specifically, they propose using a dedicated feature subspace to learn examples of biased categories appearing without their context.

Given a deep neural network, let x denote the D-dimensional output of the final pooling layer just before the fully-connected (fc) layer. Let the weight matrix associated with fc layer be $W \in R^{D \times M}$, where M denotes the number of categories given a multi-label dataset. The predicted scores inferred by a classifier (ignoring the bias term) are $\hat{y} = W^T \mathrm{x}$. To separate the feature representations of a biased category from its context, the paper does a random row-wise split of W into two disjoint subsets: W_o and W_s (dimension $\frac{D}{2} \times M$).[3] Consequently, x is split into x_o and x_s, and $\hat{y} = W_o^T \mathrm{x}_o + W_s^T \mathrm{x}_s$. When a biased category occurs without its context, the paper disables backpropagation through W_s, forcing the network to learn only through W_o, and set x_s to $\bar{\mathrm{x}}_s$ (the average of x_s over the last 10 mini-batches).

We implemented the *feature-split* method based on additional discussions with the original authors, to ensure that we replicated their method as closely as possible. For a single training batch, we first forwarded the entire batch through the model to obtain one set of scores $\hat{y}_{\text{non-exclusive}} = W_o^T \mathrm{x}_o + W_s^T \mathrm{x}_s$ and the corresponding features from the avgpool layer, which directly precedes the fc layer. We made a separate copy of these features and replaced x_s with $\bar{\mathrm{x}}_s$, then calculated a new set of output scores $\hat{y}_{\text{exclusive}} = W_o^T \mathrm{x}_o + W_s^T \bar{\mathrm{x}}_s$. Separate loss tensors for each of these outputs were computed, and elements corresponding to the exclusive and non-exclusive examples in the unmodified and modified loss tensors were zeroed out, respectively. The final loss tensor was obtained by adding these two together, and standard backpropagation was done using this final loss tensor. The gradients were calculated with respect to a weighed binary cross entropy loss:

$$L_{\text{WBCE}} = -\alpha \left[t \log(\sigma(\hat{y})) + (1 - t) \log(1 - \sigma(\hat{y})) \right], \tag{4}$$

where t is the ground-truth label, σ is the sigmoid function, and α is the ratio between the number of training images in which a biased category occurs in the presence of its context and the number of images in which it occurs in the absence of its context. A higher value of α indicates more data skewness.[4]

3.3 Strong baselines

In addition to the *standard* model, the paper compares the proposed methods with several competitive *strong baselines*.

1. **Remove co-occur labels**: For each b, remove the c label for images in which b and c co-occur.
2. **Remove co-occur images**: Remove training instances where any b and c co-occur. For COCO-Stuff, this process removes 29,332 images and leaves 53,451 images in the training set.
3. **Split-biased**: Split each b into two classes: 1) $b \setminus c$ and 2) $b \cap c$. Unlike other "stage 2" models, this model is trained from scratch rather than on top of the *standard* baseline because it has 20 additional classes. We later confirmed with the authors that they did the same.
4. **Weighted loss**: For each b, apply 10 times higher weight to the loss for class b when b occurs exclusively.
5. **Negative penalty**: For each (b, c), apply a large negative penalty to the loss for class c when b occurs exclusively. In our email communication, the authors said that the negative penalty means a 10 times higher weight to the loss.
6. **Class-balancing loss** [2]: For each b, put the images in three groups: exclusive, co-occurring, and other. The weight for each group is $(1 - \beta)/(1 - \beta^n)$ where n is the number of images for each group and β is a hyperparameter. The authors said they set $\beta = 0.99$ in our email communication.
7. **Attribute decorrelation** [5]: Use the proposed method, but replace the hand-crafted features with deep network features (conv5 features of stage 1 trained ResNet-50).

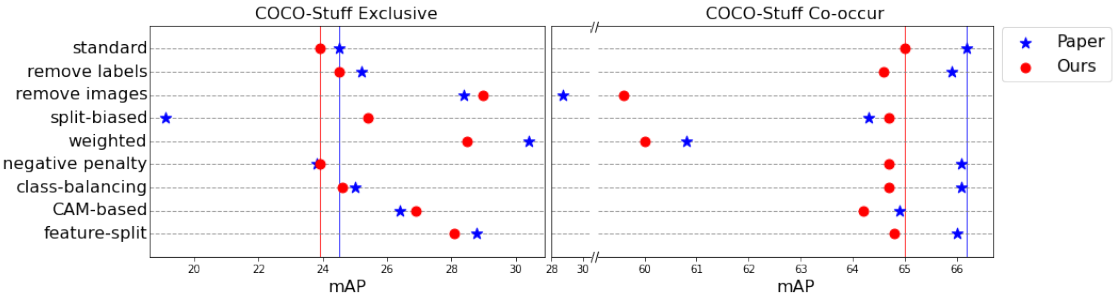
[3] In an email, the authors noted that a random split is not critical; they obtained similar results with a random split and a middle split. We observed that a middle split of W yields better results for COCO-Stuff and DeepFashion, but the opposite for AwA. As the gains from using a middle split for COCO-Stuff and DeepFashion were larger than the losses for AwA, we chose to use a middle split.

[4] In practice, the paper ensures α is at least $\alpha_{\min}$, which they set to 3 for COCO-Stuff and AwA and 5 for DeepFashion. However, we found that most of the paper's biased category pairs have α smaller than $\alpha_{\min}$. Out of 20 pairs, 13 pairs for COCO-Stuff, 20 pairs for DeepFashion, and 19 pairs for AwA had α smaller than $\alpha_{\min}$. We also tried using higher values of $\alpha_{\min}$ but didn't gain meaningful improvements, so we report results with the original authors' $\alpha_{\min}$.

Table 2: Performance of different methods on COCO-Stuff, DeepFashion, AwA, and UnRel on "exclusive" and "co-occur" distributions with best results in bold. We compare our results to the paper's results, specifically its Table 2, 3, 4, 5, 8, 9. Per-category results can be found in Section H of the appendix.

Method	COCO-Stuff (mAP)				DeepFashion (top-3 recall)				AwA (mAP)				UnRel (mAP)	
	Exclusive		Co-occur		Exclusive		Co-occur		Exclusive		Co-occur		3 categories	
	Paper	Ours	Paper	Ours	Paper	Ours	Paper	Ours	Paper	Ours	Paper	Ours	Paper	Ours
standard[5]	24.5	23.9	**66.2**	**65.0**	4.9	7.0	17.8	22.8	19.4	19.5	72.2	69.0	42.0	43.0
remove labels	25.2	24.5	65.9	64.6	6.0	7.5	**20.4**	24.4	19.1	18.9	62.9	63.2	-	42.7
remove images	28.4	**29.0**	28.7	59.6	4.2	5.6	5.4	13.0	**22.7**	**21.7**	58.3	65.2	-	48.6
split-biased	19.1	25.4	64.3	64.7	3.5	4.9	14.3	11.1	19.7	18.2	66.8	64.2	-	29.2
weighted	**30.4**	28.5	60.8	60.0	-	**29.5**	-	**43.6**	-	20.0	-	67.7	-	44.4
negative penalty	23.8	23.9	66.1	64.7	5.5	7.8	18.9	23.8	19.2	19.6	68.4	69.0	-	42.5
class-balancing	25.0	24.6	66.1	64.7	5.2	8.0	19.4	24.8	20.4	19.9	68.4	68.2	-	42.3
attribute decorr.	-	-	-	-	-	-	-	-	18.4	20.6	70.2	**69.8**	-	-
CAM-based	26.4	26.9	64.9	64.2	-	-	-	-	-	-	-	-	45.3	46.8
feature-split	28.8	28.1	66.0	64.8	**9.2**	12.2	20.1	27.1	20.8	19.2	**72.8**	68.6	**52.1**	**49.9**

Figure 2: A visual comparison of the results on COCO-Stuff from Table 2. The blue and red lines mark the paper's and our *standard* mAPs. Similar plots for DeepFashion and AwA can be found in Section F of the appendix.



3.4 Training details and computational requirements

We trained all "stage 2" models on top of the *standard* model for 20 epochs using a learning rate of 0.01, a batch size of 200, and SGD with 0.9 momentum. The exceptions are *split-biased* which is not trained on top of the *standard* model and is thus trained for an additional 20 epochs to ensure a fair comparison; and *CAM-based* which uses a batch size of 100 due to memory limits. All models were trained on a single RTX 3090 GPU and evaluated on the last epoch. On COCO-Stuff, the single-epoch training time was around 12.9 minutes for *standard*, *remove labels*, *split-biased*, *weighted*, *negative penalty*, and *class-balancing*. It took 8.4 minutes to train *remove images*, and 17.3 minutes and 13.3 minutes to train *CAM-based* and *feature-split*, respectively. Thus, we reach a different conclusion from the paper's claim that the "overall training time of both proposed methods is very close to that of a standard classifier." We suspect that this difference is due to the difference in implementation. Overall, the total training time for each method range from 35-43 hours on COCO-Stuff, 22-29 hours on DeepFashion, and 7-8 hours on AwA. For inference, the paper reports that a single forward pass of an image takes 0.2ms on a single Titan X GPU for the *standard*, *CAM-based*, and *feature-split* methods. We confirmed that it takes the same amount of time for the three methods. See Section E of the appendix for detailed training and inference times.

3.5 Results

In Table 2, we compare the performance of the ten methods, evaluated with the paper's biased category pairs for consistency. Per-category results and additional figures can be found in Sections H and F of the appendix.

COCO-Stuff: Since our *standard* model underperforms the paper's by 0.6-3.1% (Section 2), we focus on the relative ordering between the different methods visualized in Figure 2. In the paper, all but *split-biased* and *negative penalty* improve upon the *standard* baseline's "exclusive" mAP; whereas in our experiments, only *negative penalty* fails to improve on *standard*'s "exclusive" mAP. Different from the paper, *remove images* has the highest "exclusive" mAP in our experiments, followed by *weighted* and the paper's proposed methods, *feature-split* and *CAM-based*. For *feature-split*, we observed a similar tradeoff between "exclusive" and "co-occur" mAPs compared to the paper. All methods have similar performance of 55.0–55.7 mAP when evaluated on the full test set for all 171 categories.

[5]To ensure a fair comparison with the "stage 2" models, we tried training the *standard* model for an additional 20 epochs but did not see improvements; hence, we report the *standard* results from Table 1.

DeepFashion: Consistent with the paper, all methods except *remove images* and *split-biased* improve upon *standard*'s "exclusive" top-3 recall. We found the *weighted* method performs the best out of all the methods with +22.5% for "exclusive" and +20.8% for "co-occur." However, it has a relatively low top-3 recall when evaluated on the full test set for all 250 categories: 23.3 compared to other methods' top-3 recall in the range of 23.8–24.3.

Animals with Attributes: Unlike the result reported in the paper, our reproduced *feature-split* model had a -0.3% drop in "exclusive" mAP and a -0.4% drop in "co-occur" mAP compared to the *standard* model. In Section 3.6, when we experimented with different subspace sizes, we observed that the *feature-split* model trained with x_o of size 1,792 improves upon the *standard* model on both test distributions. Among all the methods, *remove images* improves the "exclusive" mAP the most; however, this method also suffers from a noticeable decrease in "co-occur" performance. When evaluated on the full test set for all 85 categories, most methods have similar mAP in the range of 72.5–73.0, except for *remove labels* that has 70.6 mAP and *remove images* that has 69.7 mAP.

UnRel: The paper includes a cross-dataset experiment where the models trained on COCO-Stuff are applied without any fine-tuning on UnRel, a dataset that contains images of objects outside of their typical context. The paper evaluates the models only on the 3 categories of UnRel that overlap with the 20 most biased categories of COCO-Stuff, which we determined to be skateboard, car, and bus. While the paper does not report results from the *remove images* baseline, for us it had the highest mAP of the 3 categories, followed by the *feature-split* and *CAM-based* methods.

3.6 Additional analyses

Cosine similarity between W_o and W_s: The paper computes the cosine similarity between W_o and W_s to investigate if they capture distinct sets of information. It reports that the proposed methods yield a lower similarity score compared to the *standard* model, and concludes that the biased class b is less dependent on c for prediction in their methods. To reproduce their results, for *feature-split*, we calculated the cosine similarity between $W_o[:, b]$ and $W_s[:, b]$ (dimensions $\frac{D}{2}$) for each b of the 20 most biased categories and reported their average. On the other hand, W_o and W_s are not specified for *standard* and *CAM-based*. Hence, we randomly row-wise split W in half and defined one as W_o and the other as W_s. In Table A6 of the appendix, we compare our reproduced results with the paper's results. While the numbers differ, our results also show that the proposed methods have weights with lower cosine similarity; thus, we conclude that we've successfully reproduced the paper's results.

Qualitative analysis: Following Section 5.1.2 of the original paper, we used CAMs to visually analyze the proposed methods. In general, our observations are in line with those of the original paper. For example, in Figure 3, we see that *CAM-based* tends to only focus on the right pixel regions (e.g., skateboard, microwave) compared to *standard*, while *feature-split* also makes use of context (e.g., person, oven). More analyses are available in Section G of the appendix.

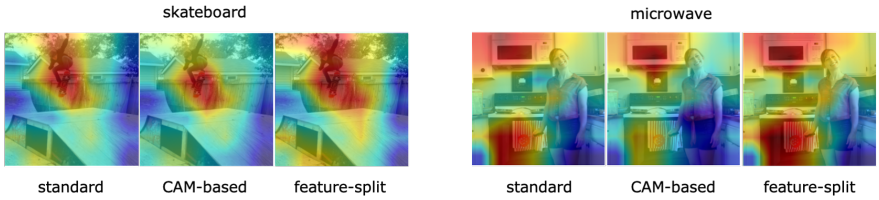


Figure 3: Biased category CAMs for (skateboard, person) and (microwave, oven) pairs.

4 Our additional experiments

What is the effect of the regularization term in the *CAM-based* method? As mentioned in Section 3.1, we tried varying the weight for the regularization term L_R (λ_2) in the *CAM-based* method that prevents the CAMs of the biased category pairs from drifting apart from the pixel regions of CAM_{pre}. We observed that weak regularization allows for highly localized, degenerate CAMs that don't resemble CAM_{pre}, while overly strong regularization makes the method less effective. We were able to strike an ideal balance with $\lambda_2 = 0.1$, higher than the paper's $\lambda_2 = 0.01$.

What is the effect of the weighted loss in the *feature-split* method? To understand the effect of the weighted loss in the *feature-split* method, we tried training a *feature-split* model without it and a baseline model with the *feature-split* weighted loss. Both variations have lower "exclusive" mAPs, suggesting that both the feature-splitting framework and the weighted loss are important components of the method. We highlight that the *feature-split* model trained without the weighted loss is worse than the *standard* model, suggesting that the weighted loss is central for the *feature-split* method to achieve good performance. However, we also observed that the *feature-split* method's weighted loss by itself is not sufficient for improving the performance of the *standard* model on the "exclusive" distribution.

Does the size of the *feature-split* subspace matter? In the *feature-split* method, the original paper allocates half of the 2,048 feature dimensions in the fc layer for learning exclusive image examples. We explored whether a smaller or larger x_o subspace may strike a better balance and improve both "exclusive" and "co-occur" performance, as the number of exclusive images is only a small fraction of the entire training data. For COCO-Stuff, the performance peaks on "exclusive" and dips on "co-occur" at the 1,024 dimension split. For DeepFashion, performance on both distributions peak at the 1,024 dimension split. For AwA, however, the performance on both distributions improves as the subspace size increases. Lastly for UnRel, the model trained on COCO-stuff with a x_o of size 768 performs best. Overall, we did not find a clear trend between *feature-split* performance and subspace size.

Table 3: (Top) Ablation studies of *CAM-based* and *feature-split* on COCO-Stuff. (Bottom) Additional *feature-split* results with varying x_o subspace sizes. Best results are in bold.

Method	Exclusive	Co-occur	All	Non-biased
standard	23.9	**65.0**	**55.7**	**72.3**
CAM-based with $\lambda_2 = 0$ (no regularization)	24.4	64.6	55.5	72.0
CAM-based with $\lambda_2 = 0.01$ (paper params)	24.6	64.6	55.5	72.0
CAM-based with $\lambda_2 = 0.1$ (tuned params)	**26.9**	64.2	55.5	72.2
feature-split	**28.1**	64.8	55.6	72.1
feature-split without weighted loss	23.6	65.4	55.6	72.1
baseline with *feature-split* weighted loss	24.0	64.8	55.5	72.1

x_o size	COCO-Stuff (mAP)		DeepFashion (top-3 recall)		AwA (mAP)		UnRel (mAP)
	Exclusive	Co-occur	Exclusive	Co-occur	Exclusive	Co-occur	3 categories
256	23.8	65.9	3.2	12.5	18.5	69.7	47.8
512	26.7	65.9	3.4	12.9	18.7	69.0	50.1
768	27.2	65.8	4.6	14.3	19.1	69.2	**50.4**
1,024	**28.1**	64.8	**12.2**	**27.1**	19.2	68.6	49.9
1,280	24.8	66.0	4.0	13.7	19.3	69.9	45.8
1,536	23.0	66.1	2.7	13.9	19.5	70.3	44.2
1,792	21.7	**66.2**	2.3	14.1	**19.7**	**70.8**	40.6

5 Discussion

We found that the proposed *CAM-based* and *feature-split* methods help mitigate contextual bias, although we could not completely replicate the quantitative results in the original paper even after completing an extensive hyperparameter search. As an effort to check our conclusions, we tried several different approaches in how we choose our best models, train the baselines, and performed evaluation. We also conducted additional analyses of the proposed methods to check our implementations and train them to achieve their best possible performance. In all cases, decreasing contextual bias frequently came with the cost of decreasing performance on non-biased categories. Ultimately, we believe deciding what method is best depends on the trade-offs a user is willing to make in a given scenario, and the original paper's proposed methods seem to strike a good balance for the tested datasets.

Recommendations for reproducibility: Overall, the paper was clearly written and it was easy to follow the explanation and reasoning of the experiments. Still, we ran into several obstacles while re-implementing the entire pipeline from scratch. Our biggest concern was making sure that most, if not all, training/evaluation details were true to the experiments in the paper. We are extremely grateful to the original authors who gave swift responses to our questions. Nevertheless, it would have been easier to reproduce the results with code or a README file listing design decisions. Given the limited information, it took us over a month to lock in various details on data processing, hyperparameter optimization, and training the *standard* model, before we could move onto reproducing the "stage 2" methods. Moreover, each method had its intricacies and we inevitably ran into ambiguities along the way. For example, the *attribute decorrelation* method took considerable time to reproduce because no hyperparameters or code were given in the paper or the original work [5]. We hope our report and published code help future use of the paper.

Recommendations for reproducing papers: In closing, we would like to share a few things that we found helpful as suggestions for future reproducibility efforts. First, writing the mandatory reproducibility plan (provided in Section I of the appendix) at the beginning of the challenge was helpful, as it forced us to define concrete steps for reproducing the experiments. We suggest putting together a similar plan because the order in which materials are presented in the paper can be different from the order in which experiments should be run. Additionally, we recommend communicating early with the original authors to determine undisclosed parameters and pin down the experimental setup. Lastly, for reproducing training processes in particular, we suggest checking how training is progressing in as many different ways as possible. In our process, this involved looking at the progression of CAMs and examining training curves for individual loss function terms, both of which helped us pinpoint our issues.

References

[1] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. COCO-Stuff: Thing and stuff classes in context. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[2] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[5] Dinesh Jayaraman, Fei Sha, and Kristen Grauman. Decorrelating semantic visual attributes by resisting the urge to share. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and Larry Zitnick. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[7] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. DeepFashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[8] Julia Peyre, Ivan Laptev, Cordelia Schmid, and Josef Sivic. Weakly-supervised learning of visual relations. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2017.

[9] Krishna Kumar Singh, Dhruv Mahajan, Kristen Grauman, Yong Jae Lee, Matt Feiszli, and Deepti Ghadiyaram. Don't judge an object by its context: Learning to overcome contextual bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[10] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(9):2251–2265, 2019.

[11] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2979–2989, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[12] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.