VB-Mitigator: An Open-source Framework for Evaluating and Advancing Visual Bias Mitigation

Ioannis Sarridis[1,2,*], Christos Koutlis[1], Symeon Papadopoulos[1] and Christos Diou[2]

[1]Information Technologies Institute, Centre for Research and Technology Hellas, 6th km Charilaou-Thermi Rd, Thessaloniki, 57001, Greece

[2]Department of Informatics and Telematics, Harokopio University of Athens Omirou 9, Tavros, 17778, Attika, Greece

Abstract

Bias in computer vision models remains a significant challenge, often resulting in unfair, unreliable, and non-generalizable AI systems. Although research into bias mitigation has intensified, progress continues to be hindered by fragmented implementations and inconsistent evaluation practices. Disparate datasets and metrics used across studies complicate reproducibility, making it difficult to fairly assess and compare the effectiveness of various approaches. To overcome these limitations, we introduce the Visual Bias Mitigator (VB-Mitigator), an open-source framework designed to streamline the development, evaluation, and comparative analysis of visual bias mitigation techniques. VB-Mitigator offers a unified research environment encompassing 12 established mitigation methods, and 7 diverse benchmark datasets. A key strength of VB-Mitigator is its extensibility, allowing for seamless integration of additional methods, datasets, metrics, and models. VB-Mitigator aims to accelerate research toward bias-aware computer vision models by serving as a foundational library for the research community to develop and assess their approaches. To this end, we also recommend best evaluation practices and provide a comprehensive performance comparison among state-of-the-art methodologies.

Keywords

AI fairness, AI bias, bias mitigation, computer vision, spurious correlations

1. Introduction

Computer vision (CV) systems have experienced significant growth and adoption across various fields [1, 2, 3]. CV advancements have substantially improved automation, efficiency, and accuracy in numerous applications. However, the widespread presence of biases in CV models remains a critical and open challenge [4, 5, 6, 7, 8]. These biases, often arising from imbalanced training datasets, cause models to learn spurious correlations rather than meaningful, generalizable patterns [9, 10, 11]. Consequently, such models often lead to unreliable predictions, reduced generalizability, and outcomes that perpetuate data bias and stereotypes. For instance, facial recognition systems trained on skewed demographic distributions have exhibited racial biases, leading to harmful real-world consequences [12, 13].

Although researchers have increasingly recognized these issues and dedicated significant effort toward bias mitigation strategies, the field suffers from fragmentation in implementation and evaluation practices, making it challenging to fairly assess and compare the efficacy of different mitigation approaches. This complicates reproducibility and slows the development of robust solutions.

To address these challenges, we introduce the Visual Bias Mitigator (VB-Mitigator), the first open-source[1] library specifically created to facilitate standardized development, evaluation, and comparative analysis of visual bias mitigation methods. VB-Mitigator provides an integrated benchmarking environment currently supporting 12 established bias mitigation approaches, 7 commonly used datasets — including synthetic datasets, datasets involving standard protected attributes (such as gender, age,

AEQUITAS 2025: Workshop on Fairness and Bias in AI | co-located with ECAI 2025, Bologna, Italy

*Corresponding author.

gsarridis@iti.gr (I. Sarridis); ckoutlis@iti.gr (C. Koutlis); papadop@iti.gr (S. Papadopoulos); cdiou@iti.gr (C. Diou)

https://gsarridis.github.io/ (I. Sarridis); https://mever.gr/author/koutlis-christos (C. Koutlis); https://mever.gr/author/papadopoulos-symeon (S. Papadopoulos); https://diou.github.io (C. Diou)

0000-0002-0064-0767 (I. Sarridis); 0000-0003-3682-408X (C. Koutlis); 0000-0002-5441-7341 (S. Papadopoulos); 0000-0002-2461-1928 (C. Diou)

[1]https://github.com/mever-team/vb-mitigator

or race), background-related biases, and general-purpose CV datasets — and evaluation metrics to comprehensively assess bias mitigation performance. A key advantage of VB-Mitigator is extensibility. Its modular architecture facilitates effortless integration of new methods, datasets, evaluation metrics, and models. Furthermore, in the context of this work, we provide an extensive comparative evaluation of the bias mitigation methods included in VB-Mitigator.

The main contributions of this work are the following:

- An open-source library designed to standardize and simplify the development, evaluation, and comparison of visual bias mitigation methods.
- A modular and extensible library architecture that allows for the easy integration of new bias mitigation methods, datasets, evaluation metrics, and models.
- An extensive comparative evaluation of 12 established bias mitigation methods.

2. Related Work

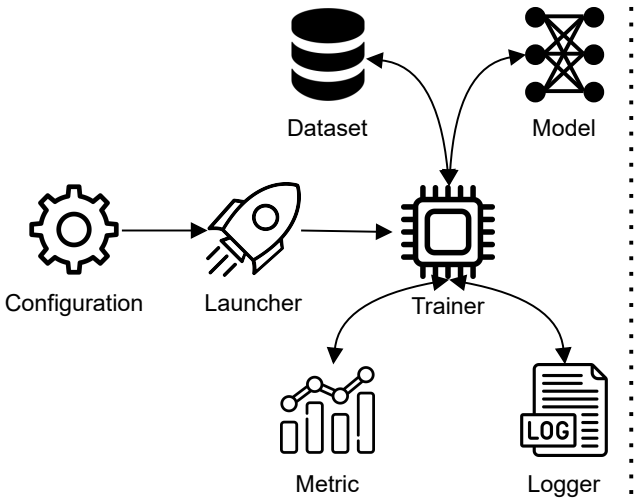
The study of bias in machine learning and computer vision has been the subject of extensive research. Surveys such as those by Mehrabi et al. [4] and Ntoutsi et al. [7] provide broad overviews of fairness challenges across AI systems, while Fabbrizzi et al. [5] analyze bias in visual datasets, and Ye et al. [9] survey focuses specifically on spurious correlations. In this work, we also focus on spurious correlations, which can either be associated with societal bias when they involve protected attributes (e.g., gender, race, or age) or represent more generic dataset biases (e.g., the tendency of waterbird images to co-occur with aquatic backgrounds [14]).

Bias mitigation methods can be broadly categorized based on their reliance on explicit bias annotations. Specifically, we distinguish between bias label unaware (BLU) methods, which operate without access to information about attributes inducing spurious correlations, and bias label aware (BLA) methods, which leverage such annotations [15, 16]. While BLA techniques often demonstrate superior performance in controlled settings, BLU approaches exhibit broader applicability and are therefore more suitable for real-world scenarios where bias annotations may be unavailable or unreliable [15].

Methods such as Group Distributionally Robust Optimization (GroupDro) [14], Domain Independent (DI) [17], Entangling and Disentangling (EnD) [18], Bias Balance (BB) [19], and Bias Addition (BAdd) [20] leverage bias annotations to guide the learning process. In particular, GroupDRO minimizes the worst-case loss across predefined groups, ensuring robust performance on minority groups. DI employs domain-specific classification heads to encourage domain-invariant representations. EnD explicitly disentangles bias representations from class representations, while BB balances bias in the logit space using prior bias knowledge. Finally, BAdd introduces bias-capturing features during training to discourage reliance on them.

A complementary line of work does not require bias labels and instead relies on bias-capturing models or other mechanisms to identify spurious correlations. This set of methods include Learning from Failure (LfF) [21], Just Train Twice (JTT) [22], Soft Contrastive (SoftCon) [19], Debiasing Alternate Networks (Debian) [23], Fairness Aware Representation Learning (FLAC) [15], and Mitigate Any Visual Bias (MAVias) [16]. Specifically, LfF reweights samples based on bias-conflicting predictions, whereas JTT focuses on misclassified samples under the assumption that they are bias-conflicting. SoftCon leverages a weighted contrastive loss derived from bias-capturing features. Debian alternates the training of main and auxiliary models to progressively reduce bias influence. FLAC minimizes the mutual information between representations and biased attributes, and MAVias uses foundation models combined with a regularization loss to infer and mitigate visual biases automatically. Some methods, such as Spectral Decouple (SD) [24], neither rely on bias labels nor auxiliary bias models. In particular, SD achieves bias robustness by regularizing the network logits, thereby reducing overfitting to spurious correlations.

Despite these advances, reproducibility and comparison across methods remain challenging. Evaluation practices are not standardized, with each study adopting its own choice of datasets and metrics. This fragmentation hinders systematic assessment. To address this gap, our work contributes an open-source



BaseTrainer	
_setup_optimizer()	_set_eval()
_setup_criterion()	_train_epoch()
_setup_scheduler()	_train_iter()
_setup_models()	_val_epoch()
_setup_device()	_val_iter()
_setup_logger()	_log_epoch()
_save_checkpoint()	_update_best()
_load_checkpoint()	_optimizer_step()
_method_specific_setups()	_loss_backward()
_metric_specific_setups()	train()
_set_train()	eval()

Figure 1: The VB-Mitigator architecture revolves around the Trainer component. The BaseTrainer class of Trainer abstracts all training pipeline steps (right), facilitating the integration of new bias mitigation methods, by re-implementing only the functions of interest.

library that integrates a diverse set of bias mitigation methods, datasets, and evaluation metrics, under a standardized environment for comparative evaluation and future extensions.

3. VB-Mitigator Architecture

Building an integrated environment for visual bias mitigation presents significant challenges. Existing techniques vary widely, intervening at diverse stages of the training pipeline—from data manipulation within data loaders and adjustments to loss functions, to the integration of external bias detection models and complex multi-stage training protocols. Compounding this, metric implementations are often dataset-specific, and limited to single or dual bias scenarios, hindering the creation of a general evaluation library. VB-Mitigator directly tackles these obstacles through an abstract and modular architecture built on PyTorch, chosen for its flexibility and robust ecosystem. By providing standardized interfaces for datasets, models, mitigation strategies, and evaluation metrics, the library enables seamless integration and comparison of diverse approaches. Overall, the design of VB-Mitigator incorporates the following key properties:

- **Modularity:** VB-Mitigator adopts a modular architecture, where each component (i.e., datasets, models, mitigators, evaluation metrics, and logging) is encapsulated in separate modules, which allows for experimentation with different configurations without altering the core functionality of the system.
- **Extensibility:** VB-Mitigator is designed to facilitate seamless integration of new bias mitigation techniques, datasets, and evaluation metrics. As previously discussed, the abstract class definitions allow for introducing new methods with minimal effort, as they only need to define the pipeline components where their approach intervenes.
- **Reproducibility:** Reproducibility in experiments is one of the objectives of VB-Mitigator. To achieve this, all operations involving stochasticity are explicitly seeded, while CUDA algorithms are configured to operate in a deterministic mode. However, complete determinism cannot be guaranteed, as it may be affected by differences in hardware and CUDA versions, as well as by certain CUDA operations that inherently lack deterministic support.

Datasets. The dataset component (`datasets/`) encapsulates PyTorch Dataset classes, engineered to return dictionaries containing input images, targets, biases (or protected attributes), and sample indices via the `__getitem__` method. Furthermore, a `builder.py` module facilitates dataset construction,

generating a comprehensive dictionary that includes critical metadata such as the number of classes, a list of protected attributes, the number of subgroups, class names, data subsets, and initial data loaders. This metadata is essential for model initialization, metric computation, training orchestration, and dynamic data loader updates.

Mitigators. The mitigator component (`mitigators/`) in VB-Mitigator acts as the core algorithmic engine, offering a flexible and standardized platform for bias mitigation algorithms. The `BaseTrainer` class establishes a comprehensive foundation for method implementation, defining functions for every stage of the training pipeline, including dataset handling, model training, metric computation, and logging (Figure 1). Additionally, the library supports method-specific configurations, facilitating the inclusion of pre-processing steps such as bias pseudo-label generation. This modular architecture allows each bias mitigation strategy to implement only the pipeline components where it actively intervenes, ensuring ease of integration and maintainability.

Models. The models component (`models/`) contains a diverse collection of neural network architectures commonly used in visual bias mitigation research. It supports a range of architectures, including lightweight Convolutional Neural Networks (CNNs) designed for small-scale datasets such as Biased-MNIST[25], widely adopted CNN architectures like ResNets [26] and EfficientNets [27], and modern vision transformers [1, 28] for more complex tasks. Additionally, it accommodates custom architectures tailored to specific bias mitigation methods, ensuring flexibility for diverse experimental setups.

Metrics. The metric component (`metrics/`) provides a comprehensive suite of evaluation metrics, tailored to bias assessment. Recognizing the common practice of employing multiple metrics for fairness evaluation (e.g., worst-group accuracy alongside average group accuracy), our implementation supports metric classes that encompass multiple measurements. Each metric class is further configured with two attributes: (i) an indicator specifying whether the metric is error-based or higher-is-better, and (ii) a designation of the primary evaluation metric that should be used for checkpoint selection.

Tools and utilities. This component (`tools/`) encompasses essential utilities for experiment management, ensuring a streamlined workflow within the framework. It includes launcher scripts for experiment execution and critical functionalities such as logging and model checkpointing. The logging system supports both Wandb[2] and TensorBoard[3] for real-time monitoring while also generating detailed logs in human-readable format and structured CSV files that can be used for visualization purposes. Additionally, it manages checkpointing, storing model states at various stages, including the latest epoch, the best-performing model, and intermediate checkpoints, enabling experiment resumption.

Configuration and execution scripts. The configuration files act as a central control mechanism, allowing researchers to seamlessly switch between datasets, bias mitigation methods, and hyperparameters. Given the extensive number of configurable variables, we utilize YACS[4], which provides a structured and efficient library to manage configurations. This enhances flexibility while maintaining clarity in experimental setups. Finally, the `scripts/` directory contains a collection of shell scripts designed to simplify and automate the execution of experiments across different bias mitigation methods and datasets.

[2] https://wandb.ai/site
[3] https://www.tensorflow.org/tensorboard
[4] https://github.com/rbgirshick/yacs

4. Methodologies

4.1. Preliminary Notation

Here, we introduce the notation used throughout the paper to ensure consistency across all methods. Let $f(\mathbf{x}; \boldsymbol{\theta})$ denote a neural network parameterized by $\boldsymbol{\theta}$, which maps an input sample $\mathbf{x} \in \mathcal{X}$ to an output prediction $\hat{y} \in \mathcal{Y}$. The dataset consists of N training samples, where each sample is associated with a target label $y \in \mathcal{Y}$ and may also include a tuple of the bias attributes labels, e.g., $a = (\textit{female}, \textit{black}, 30) \in \mathcal{A}$ for the attributes `gender`, `race`, and `age`. The learned feature representation of an input is denoted as $\mathbf{z} = h(\mathbf{x}; \boldsymbol{\theta})$, where h represents the feature extractor component of the model. The objective of a vanilla model is to learn the model parameters $\boldsymbol{\theta}$ that minimize the average loss over the training samples, formed as:

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} L(f(\mathbf{x}_i; \boldsymbol{\theta}), y_i).$$

4.2. Method Descriptions

VB-Mitigator encompasses several approaches of both categories, featuring the following BLA methods: GroupDro [14], DI [17], EnD [18], BB [19], and BAdd [20], as well as the following BLU methods: LfF [21], SD [24], JTT [22], SoftCon [19], Debian [23], FLAC (or FLAC-B) [15], and MAVias [16]. The key characteristics of these methods are reported in Table 1. Below, we briefly present the considered approaches.

Table 1
Summary of the integrated bias mitigation methods.

Method	Bias Labels	Bias-Capturing Model	Summary
GroupDRO [14]	✓	×	Minimizes worst-case loss across predefined groups.
DI [17]	✓	✓	Uses domain-specific classification heads for domain invariance.
EnD [18]	✓	×	Disentangles bias representations and entangles class representations.
BB [19]	✓	×	Balances bias in the logit space using prior bias information.
BAdd [20]	✓	✓	Adds bias-capturing features to training to discourage their use.
LfF [21]	×	✓	Reweights samples based on bias-conflicting predictions from an auxiliary model.
SD [24]	×	×	Regularizes network logits for spectral decoupling and bias robustness.
JTT [22]	×	✓	Reweights misclassified samples, assuming they are bias-conflicting.
SoftCon [19]	×	✓	Uses a weighted contrastive loss based on bias-capturing features.
Debian [23]	×	✓	Alternates training of main and auxiliary models for debiasing.
FLAC [15]	×	✓	Minimizes mutual information between representations and bias-capturing features.
MAVias [16]	×	✓	Infers and mitigates visual biases using foundation models and regularization.

Group Distributionally Robust Optimization (GroupDro). GroupDro addresses bias by minimizing the worst-case loss across predefined groups, ensuring robustness to group-level biases. Groups are defined as combinations of $\mathcal{Y}$ and $\mathcal{A}$. The objective is to minimize the maximum loss across groups, defined as:

$$\min_{\boldsymbol{\theta}} \max_{g \in \mathcal{G}} \frac{1}{N_g} \sum_{i: g_i = g} L(f(\mathbf{x}_i; \boldsymbol{\theta}), y_i)$$

where $\mathcal{G}$ is the set of groups, N_g is the number of samples in group g, and g_i is the group assignment of sample i.

Domain Independent (DI). Domain Independent (DI) aims to mitigate bias by learning representations that are invariant across different domains. Unlike traditional methods, DI employs multiple classification heads, each corresponding to a distinct domain. For each input sample $\mathbf{x}$, belonging to domain α, the model selects and utilizes only the logits produced by the classification head associated with domain α. Let $f_\alpha(\mathbf{x}; \boldsymbol{\theta})$ denote the output logits from the classification head corresponding to

domain α, where $\boldsymbol{\theta}$ represents the model parameters. The objective is to minimize the domain-specific loss while encouraging domain-invariant representations. This can be expressed as:

$$\min_{\boldsymbol{\theta}} \left[\frac{1}{N} \sum_{i=1}^{N} L(f_{\alpha_i}(\mathbf{x}_i; \boldsymbol{\theta}), y_i) \right].$$

Entangling and Disentangling (EnD). EnD explicitly disentangles representations of samples sharing the same bias label and entangles representations of samples under the same class with different bias labels. The loss function is:

$$\min_{\boldsymbol{\theta}} \Big[L(f(\mathbf{x}; \boldsymbol{\theta}), y) + \lambda_{\text{EnD},1} L_{dis}(\mathbf{z}, \alpha) + \lambda_{\text{EnD},2} L_{ent}(\mathbf{z}, \alpha, y) \Big],$$

where L_{dis} is the disentanglement loss, L_{ent} is the entanglement loss, and the regularization terms are denoted as $\lambda_{\text{EnD},1}$ and $\lambda_{\text{EnD},2}$.

Bias Balance (BB). BB infers the unbiased distribution from the skewed one and it uses a loss function that balances the impact of biases in the logit space, expressed as:

$$\min_{\boldsymbol{\theta}} L(f(\mathbf{x}; \boldsymbol{\theta}) + \mathbf{p}, y),$$

where $\mathbf{p}$ denotes the prior related to bias.

Bias Addition (BAdd). BAdd explicitly adds bias to the training procedure and encourages the model not to learn features related to that bias. The objective has the following form:

$$\min_{\boldsymbol{\theta}} L(f(\mathbf{x}; \boldsymbol{\theta}), \mathbf{b}, y),$$

where $\mathbf{b}$ denotes the bias features derived by a bias-capturing model.

Learning from Failure (LfF). LfF employs a dual-model training paradigm, simultaneously training a main classification model and an auxiliary bias prediction model. The auxiliary model is trained to predict the biased attribute $\mathcal{A}$. By comparing the losses of both models within each mini-batch, LfF identifies bias-conflicting samples. These samples are subsequently re-weighted, adjusting their impact on the primary model's training. The optimization objective is defined as:

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} w_i L(f(\mathbf{x}_i; \boldsymbol{\theta}), y_i),$$

where w_i denotes the weight assigned sample i.

Spectral Decouple (SD). SD shows that adding a regularization term to the network logits leads to a spectral decouple that enhances the network robustness on spurious correlations. The objective can be defined as:

$$\min_{\boldsymbol{\theta}} L(f(\mathbf{x}; \boldsymbol{\theta}), y) + \lambda_{\text{SD}} \|\hat{y}\|^2,$$

where λ_{SD} is the regularization weight. In contrast to other BLU methods, SD employs a generic approach to bias mitigation, operating without aiming at any bias-specific inference.

Just Train Twice (JTT). JTT focuses on learning from misclassified examples, as they tend to be bias-conflicting samples. It uses a re-weighting scheme to prioritize these examples during training by modifying the data loaders accordingly. If we denote the re-weighted samples as $\mathbf{x}'_i$ and the corresponding targets as y'_i with $i \in [0, N']$, then the loss is calculated as follows:

$$\min_{\boldsymbol{\theta}} \frac{1}{N'} \sum_{i=1}^{N'} L(f(\mathbf{x'}_i; \boldsymbol{\theta}), y'_i).$$

Soft Contrastive (SoftCon). SoftCon is a weighted SupCon [29] loss that encourages feature similarity between samples with the same target label, weighted by the cosine distance between their feature derived by a bias-capturing model and representing the attribute introducing the spurious correlation. The weights can be expressed as $w_{i,j} = 1 - \frac{\mathbf{b}_i \mathbf{b}_j}{\|\mathbf{b}_i\| \|\mathbf{b}_j\|}$ and the optimization objective is defined as:

$$\min_{\boldsymbol{\theta}} L(f(\mathbf{x}; \boldsymbol{\theta}), \mathbf{w}, y).$$

Debiasing Alternate Networks (Debian). Similarly to LfF, Debian uses an auxiliary model to encapsulate bias-related information. In particular, it employs a scheme that alternates between training a main network and an auxiliary network to mitigate bias. The predictions of the auxiliary model are used to assign weights to the main network's loss. Then, similar to LfF, the objective is:

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} w_i L(f(\mathbf{x}_i; \boldsymbol{\theta}), y_i).$$

Fairness Aware Representation Learning (FLAC). FLAC focuses on learning fair representations by minimizing the dependence between features and sensitive attributes. The objective is to minimize the mutual information between representations and sensitive attributes, defined as:

$$\min_{\boldsymbol{\theta}} L(f(\mathbf{x}; \boldsymbol{\theta}), y) + \lambda_{\text{FLAC}} I(\mathbf{z}, \alpha),$$

where λ_{FLAC} is a hyperparameter, $I(\mathbf{z}, \alpha)$ is the mutual information between representations and sensitive attributes, without accessing the $\mathcal{A}$ labels. To this end, FLAC employs a bias-capturing model trained to derive features $\mathbf{b}$. In scenarios where training a dedicated bias-capturing model is impractical (e.g., unknown biases), the biased vanilla model can be employed, resulting in the FLAC-B variant. $\boldsymbol{\theta}$ represents the learnable parameters of the main model, and $\mathbf{z}$ and $\mathbf{a}$ are vector representations of the features and sensitive attributes, respectively.

Mitigate Any Visual Bias (MAVias). MAVias employs a two-stage approach. First, it infers potential visual biases by leveraging foundational models to generate descriptive tags for input images and assess their relevance to the target class. Subsequently, these potential biases are encoded using a vision-language model and incorporated into the training procedure as regularization, discouraging the model from learning spurious correlations. The minimization objective is defined as:

$$\min_{\boldsymbol{\theta}, \boldsymbol{\phi}} L(f(\mathbf{x}; \boldsymbol{\theta}), y) + \lambda_{\text{MAVias},1} L_{reg}(\mathbf{x}, \mathbf{b}, \lambda_{\text{MAVias},2}),$$

where $\boldsymbol{\phi}$ represents the parameters of a projection layer that maps bias embeddings to the vision space, L_{reg} is a regularization term, and $\lambda_{\text{MAVias},1}$ and $\lambda_{\text{MAVias},2}$ are hyperparameters. $\boldsymbol{\theta}$ represents the learnable parameters of the main model.

Note that, although different bias mitigation methods rely on distinct loss formulations, their ultimate goal is aligned, allowing for a direct comparison of their performance on standard benchmarks in this domain.

5. Datasets

VB-Mitigator supports diverse datasets to evaluate bias mitigation techniques across a spectrum of scenarios. These datasets encompass synthetic data with controlled biases, manually injected biases in established benchmarks, and general-purpose images, facilitating a comprehensive assessment of the compared methods. The bias types considered for each dataset are those defined by the dataset providers and consistently adopted in the related literature.

Table 2
Summary of the supported datasets.

Dataset	Data Type	Bias Type	#Biases	Spurious Correlations
Biased-MNIST [25]	digits	foreground color	1	99%-99.9%
FB-Biased-MNIST [20]	digits	foreground & background color	2	90%-99%
Biased-UTKFace [19]	faces	demographics (race or age)	1	90%
Biased-CelebA [19]	faces	demographics (gender)	1	90%
Waterbirds [14]	bird species	background scene	1	95%
UrbanCars [11]	car type	background scene & co-occuring object	2	95%
ImageNet9 [30]	general purpose	background & texture	unknown	unknown

Biased-MNIST. Biased-MNIST [25], a modified version of the MNIST dataset, introduces background color correlations with digit labels. This dataset offers varying bias strengths, with 99%, 99.5%, 99.7%, and 99.9% co-occurrence levels commonly used to assess bias mitigation performance.

FB-Biased MNIST. Building upon Biased-MNIST, FB-Biased MNIST [20] introduces multiple biases by correlating both foreground and background colors with digit labels. This dataset, with suggested co-occurrence levels of 90%, 95%, and 99%, provides a more challenging benchmark.

Biased-UTKFace. The UTKFace dataset [31], comprising facial images with age, gender, and ethnicity annotations, serves as a foundation for Biased-UTKFace [19]. In this biased variant, gender is designated as the target variable, while race or age act as biasing attributes, exhibiting a 90% co-occurrence with the target. This dataset is instrumental in examining demographic biases in facial attribute classification.

Biased-CelebA. Leveraging the large-scale CelebA dataset [32], which provides annotations for diverse facial attributes, Biased-CelebA [19] focuses on gender-related biases. Here, blonde hair or heavy makeup are the target attributes, with gender demonstrating a 90% co-occurrence, enabling the study of attribute-specific gender biases.

Waterbirds. The Waterbirds dataset [14] facilitates the investigation of spurious correlations between bird species and their background environment. Featuring images of landbirds and waterbirds against corresponding terrestrial or aquatic backgrounds, it presents a 95% co-occurrence between bird species and background, serving as a benchmark for evaluating context-dependent bias mitigation.

UrbanCars. UrbanCars [11], a dataset of car images, is designed to explore biases in object recognition. It examines biases in car type classification, where background (rural or urban) and co-occurring objects introduce biases with a 95% co-occurrence rate with the target.

ImageNet-9. ImageNet-9 [30], a subset of ImageNet [33], focuses on nine object categories. It is used to investigate background-related biases in large-scale object recognition, offering a more complex and realistic evaluation scenario where biases are unknown.

The progression of datasets used in visual bias mitigation reflects an increasing complexity, mirroring the evolution of research in this domain. Early studies predominantly focused on single-attribute biases, often utilizing synthetic or simplified datasets like Biased-MNIST. Recent research has shifted towards exploring multi-attribute biases, as exemplified by FB-Biased MNIST and UrbanCars, where many existing methods struggle to achieve high performance. Furthermore, the inclusion of general CV datasets like ImageNet-9 signifies a move towards addressing the challenges of real-world scenarios, where intersectional biases may occur. Table 2 provides an overview of the considered datasets.

6. Evaluation Metrics

Evaluating the efficacy of visual bias mitigation techniques necessitates careful consideration of appropriate performance metrics. While standard accuracy, defined as:

$$\text{Acc} = \frac{|\{\mathbf{x} \in \mathcal{X} : \hat{y} = y\}|}{|\mathcal{X}|},$$

remains a foundational measure, its utility is context-dependent. Even when bias is uniformly distributed within a dataset, models may exhibit varying sensitivities to different bias attributes, rendering accuracy on a balanced test set (such as in Biased-MNIST and FB-Biased-MNIST) insufficient to capture the nuanced behavior of mitigation methods. On the other hand, in scenarios where biases are unknown and test sets are debiased through augmentations, such as by removing confounding background information in ImageNet9, accuracy constitutes a suitable measure.

Furthermore, Bias-Conflict Accuracy (BCA) is typically employed for Biased-CelebA and Biased-UTKFace datasets. BCA, defined as $\text{BCA} = \text{Acc}(\mathcal{X}_{BC})$ where $\mathcal{X}_{BC} = \{\mathbf{x} \in \mathcal{X} : a \text{ conflicts with } y\}$, attempts to focus on the underrepresented groups in the data. While this metric provides insights into performance on bias-conflicting samples, it cannot generalize to multiple, intersectional biases.

Recognizing the limitations of these metrics, we advocate for the adoption of Worst Group Accuracy (WGA) and Average Accuracy (AvgAcc). WGA, defined as $\text{WGA} = \min_{g \in \mathcal{G}} \text{Acc}(g)$, and AvgAcc, defined as $\text{AvgAcc} = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \text{Acc}(g)$, are more robust evaluation metrics, as they effectively capture performance disparities across subgroups, and are therefore more suitable for datasets with complex bias distributions and multiple bias attributes.

7. Experiments

This section presents a comparative evaluation of the methods implemented in VB-Mitigator on Biased-CelebA, Waterbirds, UrbanCars, and ImageNet9. These datasets were selected to provide a representative evaluation across diverse bias scenarios, encompassing demographic, background scene, multi-attribute, and unknown biases, respectively.

7.1. Evaluation Protocol

Given the suitability of WGA and AvgAcc for datasets with explicitly defined biases, as outlined in Section 6, these have been employed for the evaluation of models on Biased-CelebA, Waterbirds, and UrbanCars. For ImageNet9, we utilized accuracy across its seven official test set variations, which facilitate a more thorough assessment of a model's dependence on non-target object features. These variations are:

- **ORIGINAL**: The standard ImageNet9 test set, serving as the baseline.
- **ONLY-BG-B**: Images where only the background is visible, with the foreground object replaced by a black bounding box.
- **ONLY-BG-T**: Images where only the background is visible, with the foreground object replaced by an inpainted bounding box.
- **NO-FG**: Images where the foreground object has been segmented and removed.
- **ONLY-FG**: Images where only the foreground object is visible, with a black background.
- **MIXED-RAND**: Images with the foreground object placed on a random background from a random class.
- **MIXED-NEXT**: Images with the foreground object placed on a random background from the next class in the dataset.

7.2. Implementation Details

Below, we outline the data pre-processing, model architectures, and hyperparameters common to all methods, unless stated otherwise.

Biased-CelebA. We utilized the Biased-CelebA dataset with “blond hair" as the target attribute and “gender" as the spurious correlation. Images were resized to 224×224 pixels and normalized using ImageNet statistics. A ResNet18 architecture was employed, trained for 10 epochs with a batch size of 128. The Adam optimizer was used with an initial learning rate of 0.001, which was reduced by a factor of 0.1 at epochs 3 and 6. A weight decay of 0.0001 was applied.

Waterbirds. Images were resized to 256×256 pixels, center-cropped to 224×224 pixels, and normalized using ImageNet statistics. A ResNet50 model was trained for 100 epochs with a batch size of 64. The Stochastic Gradient Descent (SGD) optimizer was used with a learning rate of 0.001 and a weight decay of 0.0001.

UrbanCars. Images were resized to 256×256 pixels, center-cropped to 224×224 pixels, and subjected to random rotations (up to 45 degrees) and horizontal flips. Normalization was performed using ImageNet statistics. A ResNet50 architecture was trained using SGD for 150 epochs with a batch size of 64 and a weight decay of 0.0001.

ImageNet9. Images were resized to 256×256 pixels, center-cropped to 224×224 pixels, and normalized using ImageNet statistics. A ResNet50 model was trained for 30 epochs with a batch size of 64. The SGD optimizer was used with an initial learning rate of 0.001, which was reduced by a factor of 0.5 at epoch 25.

Method-specific hyperparameters were configured following the values recommended in their respective original publications. For GroupDro, a robust step size of 0.01 was used. In SoftCon, the Cross-Entropy loss was weighted by 0.01. The λ_{FLAC} parameter was set to 30,000, 10,000, 10,000, and 100 for Biased-CelebA, Waterbirds, UrbanCars, and ImageNet9, respectively. For JTT, bias-conflicting samples were upweighted by a factor of 100, and a learning rate of 0.00001 with a weight decay of 1 was employed. For MAVias, the $(\lambda_{\text{MAVias},1}, \lambda_{\text{MAVias},2})$ parameters were set to (0.01, 0.5), (0.05, 0.6), (0.01, 0.4), and (0.001, 0.7) for Biased-CelebA, Waterbirds, UrbanCars, and ImageNet9, respectively. For SD, the λ_{SD} parameter was set to 0.1. For EnD, the $\lambda_{\text{EnD},1}$ and $\lambda_{\text{EnD},2}$ parameters were both set to 1. Experiments were repeated for 5 random seeds on an NVIDIA A100 GPU.

7.3. Results

This section presents a comparative evaluation of bias mitigation methods across diverse datasets, encompassing varying bias scenarios. As shown in Table 3 BLA methods, such as DI and BAdd, generally exhibit superior WGA across all datasets, demonstrating the efficacy of leveraging explicit bias information. However, GroupDRO, EnD, and BB display performance variability, particularly on the UrbanCars dataset, where they all underperform. This stems from UrbanCars' multi-attribute bias structure, which contrasts with these methods' design for single-attribute biases. BLU methods, while typically achieving lower performance than BLA methods, reveal similar trends. Notably, MAVias demonstrates consistent performance across datasets, whereas SoftCon training is unstable, likely due to its strong dependence on the auxiliary model.

For ImageNet9, where biases are inherently unknown, only BLU methods are applicable. Note that here, we use the BLU variant of FLAC, denoted as FLAC-B, employing the vanilla model as the bias-capturing component. As shown in Table 4, SD and MAVias showcase robust performance across the various test set configurations of ImageNet9. Conversely, SoftCon again fails to converge.

Table 3
Worst group accuracy and average accuracy for CelebA, Waterbirds, and UrbanCars.

Method	CelebA		Waterbirds		UrbanCars	
	WG Acc (%)	Avg Acc (%)	WG Acc (%)	Avg Acc (%)	WG Acc (%)	Avg Acc (%)
GroupDRO [14]	48.88±7.6	80.76±0.9	66.38±2.2	82.80±0.8	20.00±14.2	57.76±1.7
DI [17]	84.88±1.8	91.16±0.4	91.64±0.7	94.14±0.3	**86.13**±0.4	**88.90**±0.3
EnD [18]	54.56±4.0	82.60±0.6	**94.72**±0.5	**96.10**±0.2	47.52±2.3	80.14±0.5
BB [19]	85.68±1.5	91.16±0.4	93.24±0.5	95.10±0.2	66.24±1.7	84.50±1.3
BAdd [20]	**88.98**±2.1	**91.50**±0.4	94.00±0.4	94.40±0.3	84.64±1.4	88.36±1.5
LfF [21]	58.58±6.7	82.50±2.2	94.42±0.3	96.10±0.1	46.88±2.1	80.30±0.3
SD [24]	52.10±3.5	82.50±0.7	94.10±0.4	96.24±0.2	58.56±2.7	83.54±1.2
JTT [22]	74.68±0.6	81.54±0.3	90.90±0.8	94.10±0.3	72.48±4.1	81.76±2.1
SoftCon [19]	34.34±13.8	78.34±1.8	47.04±6.2	57.20±2.1	34.72±2.8	50.46±2.0
Debian [23]	49.22±13.0	81.24±3.2	94.74±0.5	96.06±0.4	48.96±2.0	80.84±0.7
FLAC [15]	84.32±3.4	89.20±0.9	91.08±0.5	93.62±0.6	62.56±0.8	83.96±0.5
MAVias [16]	**84.88**±0.8	**90.64**±0.3	**95.90**±0.2	**96.34**±0.2	**80.80**±0.9	**87.12**±0.5

Table 4
BLU methods accuracy comparison across the 7 test sets of ImageNet9.

Method	MIXED-NEXT (↑)	MIXED-RAND (↑)	NO-FG (↓)	ONLY-BG-B (↓)	ONLY-BG-T (↓)	ONLY-FG (↑)	ORIGINAL (↑)
LfF [21]	78.70 ±0.1	81.47 ±0.2	61.07 ±0.1	34.82 ±0.2	44.46 ±0.0	88.99 ±0.2	94.34 ±0.2
JTT [22]	84.43 ±0.1	86.16 ±0.5	61.09 ±2.0	32.04 ±1.0	36.62 ±4.7	92.09 ±0.5	97.71 ±0.1
Debian [23]	83.02 ±0.4	85.64 ±0.3	64.53 ±0.4	34.45 ±0.1	45.00 ±0.6	93.06 ±0.1	97.89 ±0.1
SoftCon [19]	28.15 ±2.20	30.53 ±3.17	**28.02** ±3.05	**19.85** ±2.36	**24.07** ±2.46	33.00 ±5.63	47.47 ±4.92
SD [24]	87.56 ±0.57	88.92 ±0.74	62.60 ±1.05	31.42 ±2.93	40.81 ±3.00	**93.71** ±0.71	**98.16** ±0.06
FLAC-B [15]	84.60 ±0.46	86.62 ±0.45	59.84 ±1.67	29.71 ±0.53	40.38 ±1.28	92.72 ±0.73	97.89 ±0.09
MAVias [16]	**88.26**±0.1	**89.64**±0.2	53.02±0.7	21.83±0.4	32.48±0.6	91.90±0.4	96.92±0.2

7.4. Limitations & Future Work

While VB-Mitigator's architecture is designed to facilitate the seamless integration of existing visual bias mitigation approaches, it is important to acknowledge that future methods may introduce unforeseen integration challenges. Furthermore, it should be stressed that bias mitigation is an open research problem. The methods currently supported by VB-Mitigator, while effective across the employed datasets, do not guarantee the mitigation of bias in any possible data setup. It is worth noting that at present, VB-Mitigator focuses on in-processing methods, which attract greater research interest due to their effectiveness and broad applicability. As future work, we aim to extend its scope by also incorporating pre-processing and post-processing methodologies. Also, a key direction is integrating methods that leverage foundation models for deriving potential biases [16, 34]. Such methods will allow for evaluating fairness in a wide range of general-purpose computer vision datasets, where bias has not yet been explored, thereby enhancing this way the library's impact towards addressing bias in real-world applications.

8. Conclusion

This paper introduced VB-Mitigator, an open-source library designed to assist research in the emerging area of bias in computer vision models. The fragmentation of implementation and evaluation practices hinders progress in bias mitigation. VB-Mitigator addresses this by providing a standardized environment, promoting reproducibility and fair comparisons. The library serves as a basis for the research community, for efficient development and assessment of bias-aware approaches.

Acknowledgments

This research was supported by the EU Horizon Europe projects MAMMOth (Grant Agreement 101070285) and ELIAS (Grant Agreement 101120237).

Declaration on Generative AI

During the preparation of this work, the authors used Gemini in order to: Improve writing style. After using this service, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] A. Dosovitskiy, An image is worth 16x16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929 (2020).

[2] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: International conference on machine learning, PmLR, 2020, pp. 1597–1607.

[3] Z. Wang, Q. She, T. E. Ward, Generative adversarial networks in computer vision: A survey and taxonomy, ACM Computing Surveys (CSUR) 54 (2021) 1–38.

[4] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, A. Galstyan, A survey on bias and fairness in machine learning, ACM computing surveys (CSUR) 54 (2021) 1–35.

[5] S. Fabbrizzi, S. Papadopoulos, E. Ntoutsi, I. Kompatsiaris, A survey on bias in visual datasets, Computer Vision and Image Understanding 223 (2022) 103552.

[6] I. DeAndres-Tame, R. Tolosana, P. Melzi, R. Vera-Rodriguez, M. Kim, C. Rathgeb, X. Liu, A. Morales, J. Fierrez, J. Ortega-Garcia, et al., Frcsyn challenge at cvpr 2024: Face recognition challenge in the era of synthetic data, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 3173–3183.

[7] E. Ntoutsi, P. Fafalios, U. Gadiraju, V. Iosifidis, W. Nejdl, M.-E. Vidal, S. Ruggieri, F. Turini, S. Papadopoulos, E. Krasanakis, et al., Bias in data-driven artificial intelligence systems—an introductory survey, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 10 (2020) e1356.

[8] I. Sarridis, C. Koutlis, S. Papadopoulos, C. Diou, Facex: Understanding face attribute classifiers through summary model explanations, in: Proceedings of the 2024 International Conference on Multimedia Retrieval, 2024, pp. 758–766.

[9] W. Ye, G. Zheng, X. Cao, Y. Ma, A. Zhang, Spurious correlations in machine learning: A survey, arXiv preprint arXiv:2402.12715 (2024).

[10] C. A. Barbano, B. Dufumier, E. Tartaglione, M. Grangetto, P. Gori, Unbiased supervised contrastive learning, arXiv preprint arXiv:2211.05568 (2022).

[11] Z. Li, I. Evtimov, A. Gordo, C. Hazirbas, T. Hassner, C. C. Ferrer, C. Xu, M. Ibrahim, A whac-a-mole dilemma: Shortcuts come in multiples where mitigating one amplifies others, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 20071–20082.

[12] P. Melzi, R. Tolosana, R. Vera-Rodriguez, M. Kim, C. Rathgeb, X. Liu, I. DeAndres-Tame, A. Morales, J. Fierrez, J. Ortega-Garcia, et al., FRCSyn-onGoing: Benchmarking and comprehensive evaluation of real and synthetic data to improve face recognition systems, Information Fusion 107 (2024) 102322.

[13] I. Sarridis, C. Koutlis, S. Papadopoulos, C. Diou, Towards fair face verification: An in-depth analysis of demographic biases, in: Proceedings of the International Workshops of ECML PKDD, 2023.

[14] S. Sagawa, P. W. Koh, T. B. Hashimoto, P. Liang, Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization, arXiv preprint arXiv:1911.08731 (2019).

[15] I. Sarridis, C. Koutlis, S. Papadopoulos, C. Diou, Flac: Fairness-aware representation learning

by suppressing attribute-class associations, IEEE Transactions on Pattern Analysis and Machine Intelligence (2024).
[16] I. Sarridis, C. Koutlis, S. Papadopoulos, C. Diou, Mavias: Mitigate any visual bias, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2025.
[17] Z. Wang, K. Qinami, I. C. Karakozis, K. Genova, P. Nair, K. Hata, O. Russakovsky, Towards fairness in visual recognition: Effective strategies for bias mitigation, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 8919–8928.
[18] E. Tartaglione, C. A. Barbano, M. Grangetto, End: Entangling and disentangling deep representations for bias correction, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 13508–13517.
[19] Y. Hong, E. Yang, Unbiased classification through bias-contrastive and bias-balanced learning, Advances in Neural Information Processing Systems 34 (2021) 26449–26461.
[20] I. Sarridis, C. Koutlis, S. Papadopoulos, C. Diou, Badd: Bias mitigation through bias addition, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops, 2025.
[21] J. Nam, H. Cha, S. Ahn, J. Lee, J. Shin, Learning from failure: De-biasing classifier from biased classifier, Advances in Neural Information Processing Systems 33 (2020) 20673–20684.
[22] E. Z. Liu, B. Haghgoo, A. S. Chen, A. Raghunathan, P. W. Koh, S. Sagawa, P. Liang, C. Finn, Just train twice: Improving group robustness without training group information, in: International Conference on Machine Learning, PMLR, 2021, pp. 6781–6792.
[23] Z. Li, A. Hoogs, C. Xu, Discover and mitigate unknown biases with debiasing alternate networks, in: European Conference on Computer Vision, Springer, 2022, pp. 270–288.
[24] M. Pezeshki, O. Kaba, Y. Bengio, A. C. Courville, D. Precup, G. Lajoie, Gradient starvation: A learning proclivity in neural networks, Advances in Neural Information Processing Systems 34 (2021) 1256–1272.
[25] H. Bahng, S. Chun, S. Yun, J. Choo, S. J. Oh, Learning de-biased representations with biased representations, in: International Conference on Machine Learning, PMLR, 2020, pp. 528–539.
[26] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[27] M. Tan, Q. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: International conference on machine learning, PMLR, 2019, pp. 6105–6114.
[28] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows, in: Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 10012–10022.
[29] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, D. Krishnan, Supervised contrastive learning, Advances in neural information processing systems 33 (2020) 18661–18673.
[30] K. Y. Xiao, L. Engstrom, A. Ilyas, A. Madry, Noise or signal: The role of image backgrounds in object recognition, in: International Conference on Learning Representations, 2021.
[31] Z. Zhifei, S. Yang, Q. Hairong, Age progression/regression by conditional adversarial autoencoder, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2017.
[32] Z. Liu, P. Luo, X. Wang, X. Tang, Deep learning face attributes in the wild, in: Proceedings of International Conference on Computer Vision (ICCV), 2015.
[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 248–255.
[34] M. Ciranni, L. Molinaro, C. A. Barbano, A. Fiandrotti, V. Murino, V. P. Pastore, E. Tartaglione, Say my name: a model's bias discovery framework, arXiv preprint arXiv:2408.09570 (2024).