SAY MY NAME: A MODEL'S BIAS DISCOVERY FRAMEWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

In the last few years, due to the broad applicability of deep learning to downstream tasks and end-to-end training capabilities, increasingly more concerns about potential biases to specific, non-representative patterns have been raised. Many works focusing on unsupervised debiasing usually leverage the tendency of deep models to learn "easier" samples, for example by clustering the latent space to obtain bias pseudo-labels. However, the interpretation of such pseudo-labels is not trivial, especially for a non-expert end user, as it does not provide semantic information about the bias features. To address this issue, we introduce "Say My Name" (SaMyNa), a tool to identify semantic biases within deep models. Unlike existing methods, our approach focuses on biases learned by the model. Our text-based pipeline enhances explainability and supports debiasing efforts: applicable during either training or post-hoc validation, our method can disentangle task-related information and proposes itself as a tool to analyze biases. Evaluation on traditional benchmarks demonstrates its effectiveness in detecting biases and even disclaiming them, showcasing its broad applicability for model diagnosis. When sided with a traditional debiasing approach for bias mitigation, it can achieve state-of-the-art performance while having the advantage of associating a semantic meaning to the discovered bias.

1 INTRODUCTION

In the past decade, advances in technology have made it possible to widely use deep learning (DL) techniques, greatly impacting the computer vision field. By allowing systems to be trained end-to-end, DL offers potentially fast model deployability to solve complex problems, leading to fast progress and changing how we perceive and analyze visual information. Today, deep learning is applied in various real-world scenarios, such as self-driving cars Zhang et al. (2022a), medical imaging Yan et al. (2024), and augmented reality Liu et al. (2020).

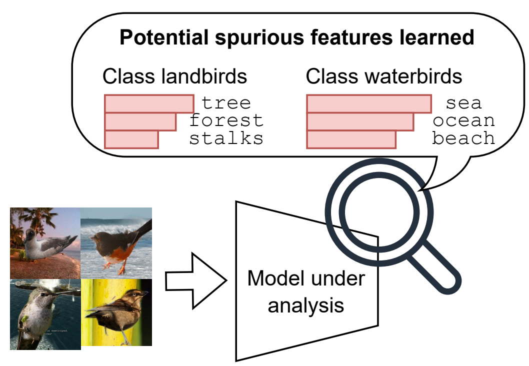


Figure 1: Our approach, SaMyNa, searches potential spurious features learned by a model, providing a ranked list of keywords.

These huge possibilities hinder potential pitfalls. One challenge to face when deploying DL solutions lies in guaranteeing that the model does not over-rely on specific patterns that are non-representative of the real-world data distribution Ming et al. (2022); Izmailov et al. (2022). This is key for safety, fairness, and ethics Tartaglione et al. (2023). To provide a simple example, when deploying solutions in autonomous driving, simple tasks like pedestrian detection might be implicitly solved by associating specific background elements (like sidewalks or pedestrian crossings) with the presence of pedestrians. This reliance on environmental cues can act as shortcuts for the network Geirhos et al. (2020), leading to poor performance when the context changes, such as when a pedestrian is crossing a road without a marked crossing lane. DL models, if not discouraged, tend to rely on spurious correlations captured at training time, especially when they are easier to learn than the actual semantic attributes Nam et al. (2020). We refer to these spurious correlations as biases, and we say that the model that learned such shortcuts is *biased* towards them.

In 2021, the European Commission introduced the Artificial Intelligence Act (AI Act) to regulate AI based on the potential risks it poses Madiega (2021). Like the General Data Protection Regulation (GDPR) GDPR (2016), the AI Act could set in the next few years a global standard. Ensuring that DL models avoid spurious biases that could affect their safety, trust, and accountability is not only essential for user safety but might soon also become a legal requirement.

A massive recent effort has been conducted by the Computer Vision community to try to discourage the presence of biases. In a nutshell, we can roughly distinguish three main research lines: (i) supervised, where the labels of the bias are provided Barbano et al. (2023); Hong & Yang (2021); (ii) bias-tailored, where a hint on what the potential bias might be is provided prior to training, and an ad-hoc model is deployed to capture it Bahng et al. (2020b); Wang et al. (2019); (iii) unsupervised, where biases are guessed directed within the vanilla-trained model Nam et al. (2020); Nahon et al. (2023); Creager et al. (2021); Li et al. (2022). When deploying a DL solution in the wild, the latter line of research appears to be the best fit, as detailed information about bias is almost surely missing. Although some solutions already exist in this context Kim et al. (2024); Nam et al. (2020); Nahon et al. (2023); Ji et al. (2019), there is still a gap in the literature related to the problem of naming a specific bias affecting a DL model, providing natural language descriptors that can be directly interpretable by a human. Existing solutions either start from a predefined set of attributes Eyuboglu et al. (2022); Wiles et al. (2023) or explicitly require the availability of a validation set known to contain bias-conflicting samples Kim et al. (2024), and requires to perform computationally expensive operations such as captioning the entire validation set, which may be unavailable or made up of only aligned samples. Our method is effective by using only the training set, relaxing the assumptions of existing works (e.g. Kim et al. (2024)), thus setting a first step towards mining specific model biases in realistic scenarios. Unlike approaches that discover biases in the dataset, our focus is oriented on naming the biases captured by the model under exam. We mine the specific features in common with these samples and we associate semantic (textual) meaning to them. From this, an expert user (or prospectively a certifier software) can search and discriminate whether the learned feature is a bias or rather a feature for the system (Fig. 1). Through "Say My Name" (SaMyNa), we aim at providing a tool that enhances explainability for the DL model's learned features, on top of which, if necessary, any state-of-the-art debiasing approach can be used to sanitize the model.

Our contributions are here summarized at a glance.

- We propose a tool able to potentially give a name to biases in DL models. We propose a pipeline where the whole process is text-based; contrary to end-to-end approaches, all the intermediate steps of our approach are humanly readable and interpretable (Sec. 3.2).
- Within our approach, we can focus on potential biases by disentangling specific work domains where the model is also tested by using the embedding space of a text encoder (Sec. 3.2.4).
- Our approach is usable both at training and at inference time. For the former, we propose a simple yet effective strategy to mine biases directly on the training set (Sec. 3.1).
- We test our approach on well-known setups, finding the biases acknowledged by the community (Sec. 4.3.1). Besides, we also test on ImageNet-A, distinguishing cases where a bias exists and where, on the contrary, the issue is not bias-related (Sec. 4.3.2). This demonstrates the broad applicability of SaMyNa even for more general DL model diagnosis, and after bias discovery (Sec. 3.1) and naming (Sec. 3.2) when coupled with a standard debiasing strategy, can attain debiasing results in line with the state-of-the-art (Sec. 4.4), with the big plus of assigning semantic meaning to the discovered bias.

2 Related Works

The problem of bias and model debiasing has been widely explored in recent years, within three main frameworks, differing from how or if bias knowledge is explored for mitigating model dependency on bias: supervised, bias-tailored, and unsupervised.

Supervised Debiasing. Supervised methods require explicit knowledge of the bias, generally in the form of labels, indicating whether a sample presents a certain bias or not. One of the most typical approaches consists of training an explicit bias classifier, trained on the same representation space as the target classifier, in an adversarial way, forcing the encoder to extract unbiased representations Alvi et al. (2018); Xie et al. (2017). Alternatively, bias labels can be exploited to identify pre-defined groups, training a model to minimize the worst-case training loss, thus pushing the model towards learning biased samples Sagawa* et al. (2020). Another possibility is represented by regularization terms, which aim at achieving invariance to bias features Barbano et al. (2023); Tartaglione et al. (2021).

Bias-Tailored Debiasing. Bias-tailored approaches usually rely on some kind of knowledge about the bias nature. For example, if the bias is textural, then custom architectures can be designed to be more sensitive to textural information. For example, Bahng et al. (2020a) propose ReBias, where a custom texture bias-capturing model is designed using 1x1 convolutions. A similar approach is followed by Hong & Yang (2021), where a BagNet-18 Brendel & Bethge (2019) is used as a bias-capturing model.

Unsupervised Debiasing. Differently from the previously described approaches, unsupervised debiasing methods do not assume any prior knowledge of the bias, facing a more realistic situation where bias is unknown. Nam *et al.* Nam et al. (2020) propose LfF, where a vanilla bias-capturing model is trained with a focus on easier samples (bias-aligned), using the Generalized Cross-Entropy (GCE) loss Zhang & Sabuncu (2018), while a debiased network is trained by giving more importance to the samples that the bias-capturing model struggles to discriminate. Ji *et al.* Ji et al. (2019) propose an unsupervised clustering method that learns representations invariant to some unknown or "distractor" classes in the data, by employing over-clustering. A set of unsupervised methods relies on the assumption that bias-conflicting samples are likely to be misclassified by a biased model Kim et al. (2022a); Liu et al. (2021). In Liu et al. (2021) a model is trained for a few epochs and then used in inference on the training set, considering misclassified samples as bias-conflicting and vice-versa. The debiasing is then performed by up-sampling the predicted bias-conflicting samples. In Kim et al. (2022a) the training set is split into a fixed number of subsets, training a model on each of them. Then, the trained models are ensembled into a *bias-commmittee* and the entire training set is fed to the committee, proposing that debiasing can be performed using a weighted ERM, where the weights are proportional to the number of models in the ensemble misclassifying a certain sample. Similarly to Nahon et al. (2023), we are able to identify during the training of a vanilla model in which moment the bias is potentially best fitted by the model, with the advantage of working directly at the output of the same model instead of mining the information in its latent space. This comes both with computational advantages (given that the latent space is typically higher dimensional) and with better interpretability of the outcome, given that we work in the model's output space.

Bias Naming. Recently, methods exploiting natural language and vision-language models to identify and mitigate bias have been proposed Eyuboglu et al. (2022); Kim et al. (2024); Zhang et al. (2023). Zhang *et al.* Zhang et al. (2023) introduce a method capable of determining subsets of images with similar attributes systematically misclassified by a model (i.e., error slices) and a rectification method based on language. However, it starts from a pre-defined set of attributes, thus hindering the possibility of discovering completely unknown and multiple biases. Eyuboglu et al. (2022) exploits a cross-modal embedding space to identify error slices, providing natural language predictions of the identified slices. Wiles *et al.* Wiles et al. (2023) propose a method for automatically determining a model's failures, exploiting large-scale vision-language models and captioners to provide interpretable descriptors of such failures in natural language. In Kim et al. (2024), the authors propose to extract class-wise keywords representative of bias, later used for model debiasing, exploiting group-DRO Sagawa* et al. (2020) on the identified groups. In this work, a CLIP score is defined using the similarity between extracted keywords and correctly and misclassified samples (class-wise) and used to find the keywords associated with a bias. In D'Incà et al. (2024), the authors use large-language models and text prompts for bias discovery in text-to-image generative models. Differently from the previously cited works, we introduce an unsupervised method for diagnosing a model dependency on bias for image classification tasks, that can either be performed during or after training and only exploits task-related knowledge to provide a transparent analysis on the potential hidden biases captured by the model.

3 METHOD

In this section, we present our proposed method SaMyNa to identify potential spurious correlations learned by the model under analysis (Sec. 3.2). Our method can be plugged to perform model diagnosis either at training time or at test: for the first case, we also present an approach to identify at which point to mine a potential bias for the target model (Sec. 3.1).

3.1 MINING MODEL'S BIASES

Consider a supervised classification setup (having C target classes), where we learn from a dataset $\mathcal{D}_{\text{train}}$ containing N input samples $(\boldsymbol{x}_1, \dots, \boldsymbol{x}_N) \in \mathcal{X}$, each with a corresponding ground truth label $(\hat{y}_1, \dots, \hat{y}_N) \in \mathcal{Y}$. A deep neural network $\mathcal{M}$, trained for t iterations, produces an output distribution $\boldsymbol{y}_{t,n} \in \mathbb{R}^C$ over the C classes, for each input $\boldsymbol{x}_n$ (typically, the activation of the last layer is a softmax). The network is trained to match the ground truth label $\hat{y}_n$ by minimizing a loss function such as cross-entropy.

If any bias is present in the training set, however, the learning process could drive the model towards the selection of spurious features Sagawa* et al. (2020); Nam et al. (2020); Bahng et al. (2020a), resulting in misclassification errors. Recent findings show that it is possible to identify the moment when the model best fits the bias in the training set Nahon et al. (2023). We will formulate here the problem of identifying, at training time, when the trained model maximally fits a potential bias.

We say that $\mathcal{M}$ misclassifies the n-th sample at the t-th learning iteration if $\hat{y}_n \neq \arg\max(\boldsymbol{y}_{t,n})$. Focusing on this example, we know that by minimizing the loss function we aim at increasing the value of the $\hat{y}_n$-th component $\boldsymbol{y}_{t,n}(\hat{y}_n)$ (while decreasing all the others). Inspired by the Hinge loss function Rosasco et al. (2004), we can define a per-sample distance metric telling us how far the n-th sample is from being correctly classified:

$$d_{t,n} = \begin{cases} \max(\boldsymbol{y}_{t,n}) - \boldsymbol{y}_{t,n}(\hat{y}_n) & \text{if} \arg\max_n(\boldsymbol{y}_{t,n}) \neq \hat{y}_n \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Intuitively, the higher equation 1 is, the most $\mathcal{M}_t$ is confident in misclassifying n. We are interested in finding the iteration t^* such that the model most confidently misclassifies a pool of samples:

$$t^* = \arg\max_t \frac{1}{\sum_n \bar{\delta}[\hat{y}_n - \arg\max_n(\boldsymbol{y}_{t,n})]} \sum_n d_{t,n}, \tag{2}$$

where δ is the Kronecker delta and $\bar{\delta} = 1 - \delta$. When reaching t^*, the most informative samples are the misclassified ones: given that the model is most confident in misclassifying them, then the model has clearly learned some spurious features. Differently from prior works Nahon et al. (2023) speculating that misclassified samples embody a bias (with the goal of applying debiasing methods), our goal is to understand why these samples are misclassified, ultimately providing an end user of the system the possibility to acknowledge the presence of a bias. For the model $\mathcal{M}_t$ we will split the training dataset in a pool of correctly classified samples $\mathcal{D}_{\text{train}}^{\text{correct}}$ and misclassified $\mathcal{D}_{\text{train}}^{\text{misclass}}$. Examples of vanilla model's output softmax distribution during training can be found in the Supp. Material.

3.2 BIAS NAMING

We present here SaMyNa, our bias naming approach starting from a trained model $\mathcal{M}$ from which we aim to run our bias naming tool. Fig. 2 proposes an overview of the main pipeline we used, consisting of the following steps:

1. *Samples subset selection.* Given that the objective of our proposed method is to identify biases learned by $\mathcal{M}$, we are allowed to propose a subset of most representative samples for a given target class (Sec. 3.2.1).
2. *Samples captioning.* Once the samples are selected, a multimodal LLM captioning tool is used to extract a textual description of each sample (Sec. 3.2.2).
3. *Keywords selection.* Starting from the computed captions, we mine keywords in common from the textual description of the samples within the same learned class (Sec. 3.2.3).

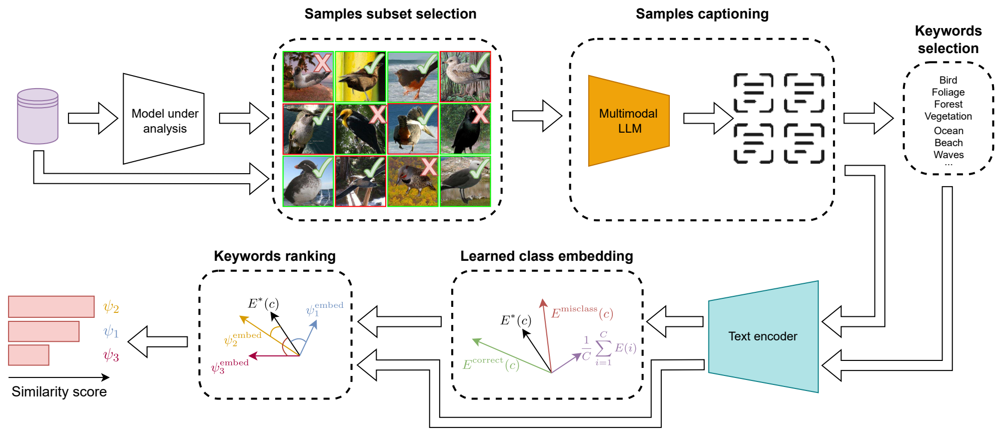


Figure 2: Pipeline for SaMyNa. Given a model, we can tell on either $\mathcal{D}^{\text{train}}$or $\mathcal{D}^{\text{val}}$ what are the correctly (with green border) and the incorrectly (red border) classified samples. Amongst these, we first perform a sample subset selection looking at the latent space of the model under analysis and choosing through k-medoids the most representative samples for the learned class. Then, we employ a captioner to get a textual description of these samples. Among these descriptions, we identify recurrent keywords and, in parallel, working in the latent space of a text encoder, we extract the mean description for the learned classes, cleansed from common features within the dataset. We finally compare this representation with the embedding of the keywords, identifying learned correlations aside from the target.

4. *Learned class embedding.* Starting from a textual description of the samples, we can extract the shared information between the correctly classified samples and the incorrectly classified ones: this will constitute the embedding for a potential bias (Sec. 3.2.4).
5. *Keywords ranking.* The embedding of the learned class is compared with the embedding of recurrent keywords in the captions and we get a ranking for the keywords most aligned with the learned class (Sec. 3.2.5).

3.2.1 Samples subset selection

Given $\mathcal{M}$, for a given target class c, we extract the pool of correctly classified samples $\mathcal{D}^{\text{correct}}(c)$ and samples misclassified as c $\mathcal{D}^{\text{misclass}}(c)$.[1] Provided that $\mathcal{M}$ clusters both $\mathcal{D}^{\text{correct}}(c)$ and $\mathcal{D}^{\text{misclass}}(c)$ together, our hypothesis is that these two share a common set of features, behind which we might find a bias. In the typical deployment scenario, the correctly classified examples are abundant, and, for instance, $\mathcal{M}$ projects them in a very narrow neighborhood of its latent space. We build on top of this observation and run a k-medoid algorithm to reduce the cardinality of correctly classified samples. Our long-range objective will be indeed to capture the set of features that are correctly learned by the model, and k-medoids is a natural choice to have a good coverage of the latent space for $\mathcal{M}$.

3.2.2 Samples captioning

At this point, we will generate captions from the selected samples. To do this, we use a pre-trained multimodal large language model that takes as input both a prompt and an image. The choice of the prompt is kept generic and asks to generate a textual description of the content of the image, providing some context of the target task. We decided to employ a large-scale image captioner in all our experiments, given that biases might hide in the subtle characteristics of the provided images.

3.2.3 Keywords selection

From the captions obtained in the previous step (Sec. 3.2.2), we select recurrent keywords. First, we perform some NLP standard processing, consisting of lower-case conversion, remotion of non-letter

[1]please note that we have dropped the “train” subscript from the $\mathcal{D}$ partitioning as this pipeline will work equally well also when using a validation set.

or digit characters, and stop-words removal. Each caption is word-level tokenized Bird et al. (2009), and every token is considered a potential keyword. Then, we count the frequencies of the obtained keywords within the same class c. Lastly, we filter out the keywords appearing in the captions of less than the f_{min} fraction of samples. The remaining keywords are aggregated and constitute a keywords proposal pool Ψ.

3.2.4 LEARNED CLASS EMBEDDING

In parallel to keywords selection, we aim at having a representation for the learned class, disentangled from the specific domain $\mathcal{M}$ is trained to. To do this, we work in the embedding space of a pre-trained text encoder. From the generated captions we obtain, for each class c, the embedding matrices $E^{\text{correct}}(c) \in \mathbb{R}^{|\mathcal{D}^{\text{correct}}(c)|\times Z}$ and $E^{\text{misclass}}(c) \in \mathbb{R}^{|\mathcal{D}^{\text{misclass}}(c)|\times Z}$, where Z is the dimensionality of the embedding vectors. Our goal here is to calculate the embeddings $E^*(c) \in \mathbb{R}^Z$ semantically representing the learned representation of the class c. Not only that, we would like to disentangle this from the features in common to all the classes learned from the model, given that $\mathcal{M}$ could be trained (and tested) to fit a specific domain, which in such a specific case would not constitute a bias but rather a feature. For this, we first calculate the average embedding $E(c)$ for a specific learned class:

$$E(c) = \frac{\sum_{i=1}^{|\mathcal{D}^{\text{correct}}(c)|} \sum_{j=1}^{|\mathcal{D}^{\text{misclass}}(c)|} [E_i^{\text{correct}}(c) + E_j^{\text{misclass}}(c)]}{2\left[|\mathcal{D}^{\text{correct}}(c)| \cdot |\mathcal{D}^{\text{misclass}}(c)|\right]}. \tag{3}$$

$E(c)$ will now contain all the common features of the class c. However, it will also contain some information shared in the entire dataset (for example common characteristics of the different classes). From this, we can extract the embedding without the shared information from the dataset through:

$$E^*(c) = E(c) - \frac{1}{C}\sum_{i=1}^{C} E(i). \tag{4}$$

The intuition of this approach originates from the arithmetic and semantic properties of natural language latent spaces Mikolov et al. (2013). To provide a realistic example, consider the task of gender recognition from facial pictures, in which the hair color is a spurious correlation. $E(c)$ might contain features related to concepts such as “blonde” and “face”. As we are only interested in the former, computing $E^*(c)$ is an effective solution to filter out the shared information “face”.

3.2.5 KEYWORDS RANKING

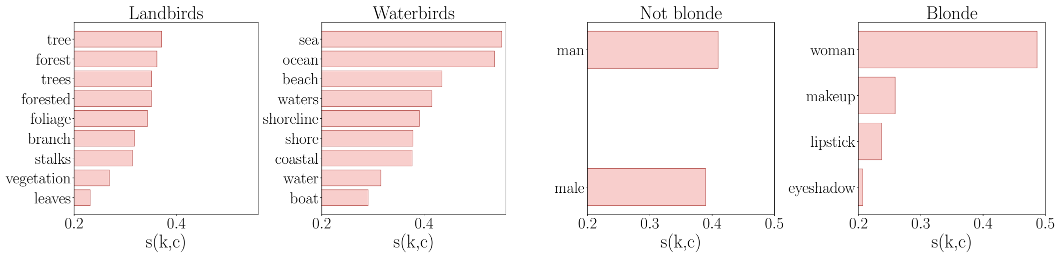
Now, we are ready to compare the embedding of each keyword with $E^*(c)$ using the cosine similarity:

$$s(\psi, c) = \text{sim}[\psi^{\text{embed}}, E^*(c)], \quad \psi \in \Psi, \tag{5}$$

where ψ^{embed} is the embedding of the keyword in the same latent space used to calculate $E^*(c)$. This tells us how much the concept is embodied by the proposed keywords. Based on the ranking, we will obtain a set of keywords that correlate with the learned class c, and others that become decorrelated as they embody some knowledge shared through all the classes (as filtered in equation 4). For this, we introduce a hyper-parameter $t_{\text{sim}} > 0$ that thresholds the relevant keywords for the learned class c, based on the similarity score. Finally, as post-processing, we filter all the keywords related to the ground-truth target class the model was aiming at learning: the final ranking we obtain embodies the set of features that correlate with the learned class c, from which an end user of the system can deduce the presence of a bias.

4 EMPIRICAL RESULTS

We provide here the main results obtained. We highlight that, for visualization purposes, all the figures contain up to the top nine keywords identified by SaMyNa: the full results, along with the ablation study, are presented in the Supplementary material. For our experiments, we have employed an NVIDIA A5000 with 24GB of VRAM, except for the captioning step for which we have employed an NVIDIA A100 equipped with 80GB of VRAM. The source code, attached to the submission, will be open-sourced upon acceptance of the article.



(a) (b)

Figure 3: Similarity scores for Waterbirds (a) and CelebA (trained on the *blonde hair* attribute) (b).

4.1 Setup

Models tested. We tested the most popular architectures benchmarked from the debiasing literature: ResNet-18 for CelebA and BAR, and ResNet-50 for Waterbirds and ImageNet-A. All the models are pre-trained on ImageNet-1K, with architecture and weights provided by `torchvision`. On ImageNet-A, models are run only in inference, as we are interested in mining biases already existing in the original pre-trained models, while for all the other experiments we apply the training procedure described in Sec. 3.1. For this step, we train with a batch size of 128 and a learning rate of 0.001 for Waterbirds, as done in Sagawa* et al. (2020); for CelebA, we use a batch size of 256 and a learning rate of 0.0001, following Nam et al. (2020). For both, we employ SGD with Nesterov momentum set to 0.9. Finally, for BAR, we employ a batch size of 256 and a learning rate of 0.001, with Adam as optimizer Kim et al. (2021).
Captioning. For the captioning model, we used LLaVA-NeXT Liu et al. (2024)[2] in its 34B configuration, quantized in 8 bits. Before feeding our input images to the image captioner, we apply the default corresponding preprocessing transform, as provided by the huggingface library. The prompts we used to generate the captions are personalized for each dataset and their length is limited to 300 tokens. All the employed captions can be consulted in the Supplementary material.
Class and keywords embedding. For this part, we used the Sentence-BERT model[3] from the sentence-transformers Reimers & Gurevych (2019) library to generate 384-dimensional embeddings of the captions. The minimum frequency for the keywords f_{min} is 15%. For the correctly classified samples, we set $k = 10$ and t_{sim} is set to 0.2.

4.2 Datasets

Before we present the results obtained with our bias naming pipeline, we briefly describe the datasets employed in this study: Waterbirds Sagawa* et al. (2020), CelebA Liu et al. (2015), BAR Nam et al. (2020), and ImageNet-A Hendrycks et al. (2021).
Waterbirds. Waterbirds is an image dataset introduced in Sagawa* et al. (2020) to test the robustness of optimization methods against distribution shifts. The associated task is to classify the habitat of bird species, divided into *waterbirds* and *landbirds*. In the training set, though, 95% of waterbirds are associated with a water background, and 95% of landbirds are set on land. Only 5% of the two classes' samples are presented with an *opposite* background. This results in a potentially strong spurious relation between the target label and the background.
CelebA. CelebA Liu et al. (2015) is one of the most popular datasets of face images, depicting celebrity individuals with multiple samples per subject and equipped with extensive annotations regarding roughly 40 attributes, encompassing several face and style characteristics. Regardless of its popularity, it is notoriously affected by several biases Nam et al. (2020); Barbano et al. (2023). In this work, we analyze the task of classifying whether a person has blond hair or not, for which the attribute *female* is not uniformly represented, but spuriously correlated with the *blond* class.
BAR. *Biased Action Recognition* (BAR) is an action recognition dataset crafted by Nam *et al.* in Nam et al. (2020). The target classes of BAR (*Climbing*, *Diving*, *Fishing*, *Racing*, *Throwing*, *Vaulting*) present a strong bias towards the setting where they are performed. For instance, the large

[2] `https://huggingface.co/llava-hf/llava-v1.6-34b-hf`
[3] `https://huggingface.co/sentence-transformers/paraphrase-MiniLM-L12-v2`

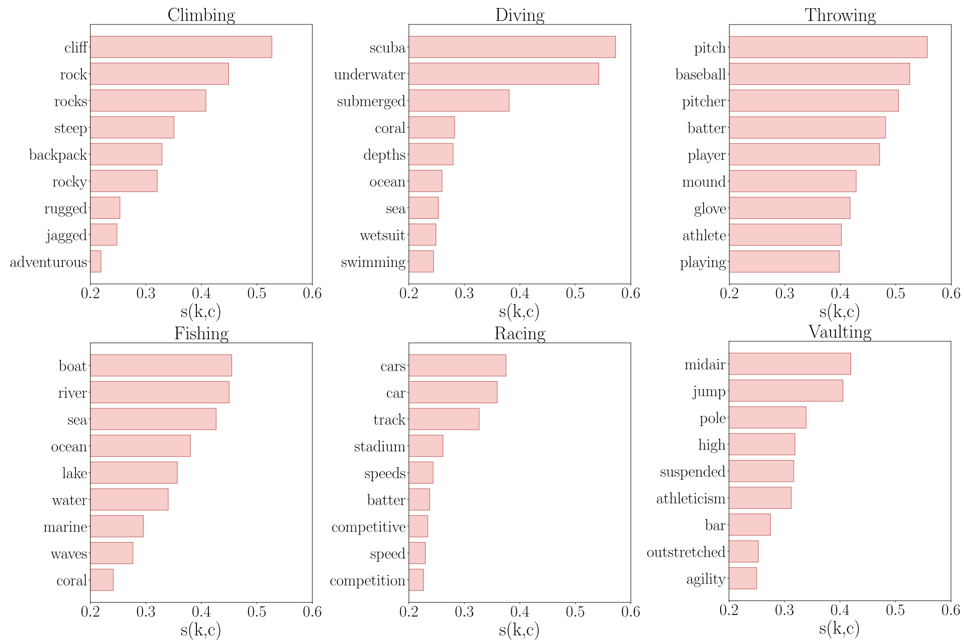


Figure 4: Similarity scores for the BAR dataset.

majority of samples from the class *Racing* is depicted in a circuit track context, while in the test set, we can find many *off-road* settings on which deep classification models fail to generalize. This dataset does not provide bias group annotations and does not provide a proper validation set, making impossible to compute a Worst Group accuracy.

ImageNet-A. ImageNet-A is a collection of 7,500 real-world images sharing the same category of a 200-class subset of ImageNet, onto which deep models systematically fail to output the correct prediction. Originally introduced in Hendrycks et al. (2021) for testing adversarial model robustness, it is also commonly adopted in the context of model bias Bahng et al. (2020b); Kim et al. (2022b).

4.3 Bias Discovery

Our empirical validation on bias discovery is here divided into naming the bis during training (Sec. 4.3.1) and naming it post-hoc, at inference (Sec. 4.3.2). In Sec.4.4 we provide a description of how our keywords can be used to perform bias mitigation.

4.3.1 Naming the bias at training time

We begin by discussing the results of our bias naming pipeline when applied in a model's training process on Waterbirds, CelebA, and BAR.

Waterbirds. The barplot in Fig. 3a shows the candidate bias keywords for Waterbirds, alongside their relative similarity value. We can observe how the obtained keywords are mainly related to the background information (`tree` and `forest` for landbirds; `sea` and `ocean` for waterbirds). Most importantly, the top keywords display high similarity values, indicating a high correlation with their class targets. We deduce that the model suffers from a bias about image backgrounds, which indeed is the case for Waterbirds. It is also worth noticing how the top similarities differ among the two classes. We hypothesize two possible factors causing it: (i) model bias towards `sea` is stronger than the one towards `tree`, as it is constituted by simpler visual patterns, easier to learn for the network; (ii) the *landbirds* class has a much larger population, thus allowing for the bias on this class to be averaged over more instances than the *waterbirds* case.

CelebA. In analyzing the possible biases of our vanilla model on CelebA (see Fig. 3b), we find that the top-1 keyword for both classes represents a gender: `male/man` for class *not blonde* (with similarity ≈ 0.4), and `woman` for class *blonde* (similarity of 0.49). Additionally, we find among the *blonde* class, keyword terms typically associated with gender stereotypes, such as `makeup` and

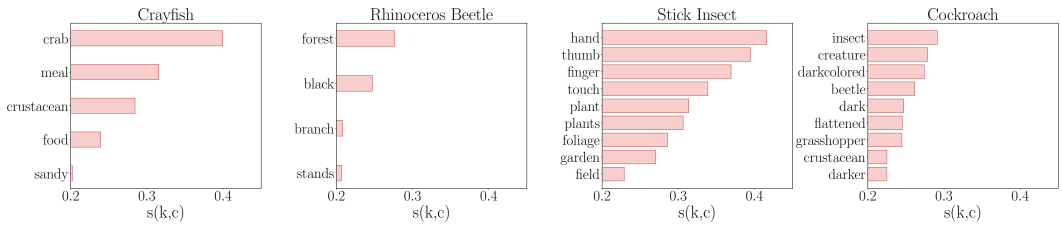


Figure 5: Similarity scores for the *crayfish*, *rhinoceros beetle*, *stick insect* and *cockroach* classes from ImageNet-A.

`lipstick`, with moderately high similarity. This suggests that the vanilla model is not only biased regarding the classification task but also incorporates *unfair features*, arising from societal biases reflected in the training data.

BAR. Bar presents a more challenging situation, due to the complete absence of bias group annotations. Regardless, the output of our approach still provides insights about the biases captured by the model. In the training class *climbing*, scenes where the subject is ascending on rocks, are overly represented, and thus the vanilla model has wrongfully learned to rely on the presence of rocky backgrounds. This is reflected by the top three keywords in Fig. 4 (`cliff`, `rock` and `rocks`), all having similarities above 0.4. Similar considerations can be made on the other classes: `pitch`, `batter`, `glove` suggest an even stronger bias for the class *throwing* towards a specific sport, baseball, which is also the second most correlated keyword. The same can be said for `scuba` in the *diving* class. Keywords for other classes show slightly lower maxima. Still, we can measure relevant correlations with the presence of cars and circuit races for *racing*, as well as boats and water (e.g. `rivers`, `sea`) for *fishing*. Complete outputs of the keyword extraction process are available in the Supp. Material.

4.3.2 Naming the bias at inference time

To evaluate the capabilities of our method in describing potential biases at inference time, we design a dedicated experiment involving ImageNet-A. In particular, we are interested in finding specific model failures and extracting an interpretable set of keywords that can guide an expert practitioner to tackle them. With this aim, we first build an evaluation set as the union of the whole ImageNet-A dataset with the samples in the validation set of ImageNet-1K sharing the same 200 categories of ImageNet-A. Then, we run the model in inference over this dataset, collecting $\mathcal{D}^{\text{correct}}$ and $\mathcal{D}^{\text{misclass}}$ directly without any additional training. In this experiment, we are not interested in finding dataset-wise biases, but rather in assessing the behavior of our approach in specific and challenging real-world scenarios. Hence, we derive a specific case study from the systematic model confusion obtained from its predictions, which could hide the presence of a possible model bias. Another analysis describing more general DL diagnosing features is provided in the Supplementary Material, while here we limit the discussion to the model bias perspective.

Our key study (see Fig. 5) involves a subset of four classes: *crayfish*, *rhinoceros beetle*, *stick insect*, and *cockroach*. Samples (correctly or incorrectly) classified as *crayfish* are often placed in a setting depicting plates, tables, or people eating, thus the bias reflected by the keywords `meal` and `food`. Moreover, `crab` probably suggests the inability of the model to distinguish between the two closely related species. At the same time, for the classes *stick insect* and *rhinoceros beetle*, several samples show the creature being held in a person's hand, often in a vegetation setting (hence the keywords `hand`, `thumb`, ..., `plants`, `foliage`). From these observations, we validate the presence of a possible bias among these classes, for which the source of error is not caused only by the fine-grained nature of these categories. Additional insights from our analysis of ImageNet categories can be found in the Supp. Material, including some analysis involving transformer-based Vision Models.

4.4 Mitigating the Discovered Bias

Starting from the keywords that were extracted to describe potential bias affecting the classification model, we here validate if the attributes suggested by SaMyNa can be leveraged to improve our network's generalization capabilities.

Setup. After bias identification and naming (SaMyNa), we exploit a pre-trained CLIP model as a

Method	No Val.Set in Bias-Id	Unsup.	Bias Named	CelebA (Hair Color)		Waterbirds		BAR
				Average	Worst Group	Average	Worst Group	Average
LISA Yao et al. (2022)	–	✗	✗	92.40	89.30	91.80	89.20	–
GroupDRO Sagawa* et al. (2020)	–	✗	✗	92.90 ± 0.20	88.90 ± 2.30	93.50 ± 0.30	91.40 ± 1.10	–
George Sohoni et al. (2020)	✗	✓	✗	94.60	54.90 ± 1.90	95.70	76.20 ± 2.00	–
JTT Liu et al. (2021)	✗	✓	✗	88.10	81.50 ± 1.70	89.30	83.80 ± 1.20	68.53 ± 3.29
CNC Zhang et al. (2022b)	✗	✓	✗	89.90	88.80 ± 0.90	90.90	88.50 ± 0.30	–
B2T+GDRO Kim et al. (2024)	✗	✓	✓	93.20	90.40 ± 0.90	91.50	$\mathbf{90.70 \pm 0.30}$	–
ERM	✓	✓	✗	**94.90**	47.70 ± 2.10	**97.30**	62.60 ± 0.30	51.85 ± 5.92
LfF Nam et al. (2020)	✓	✓	✗	84.24	81.24 ± 1.38	91.20	78.00	62.98 ± 2.76
DebiAN Li et al. (2022)	✓	✓	✗	84.00	52.90 ± 4.70	–	–	69.88 ± 2.92
SaMyNa (ours) + GDRO	✓	✓	✓	92.20 ± 0.01	$\mathbf{90.60 \pm 0.08}$	91.20 ± 0.04	$\mathbf{90.70 \pm 0.01}$	$\mathbf{71.30 \pm 0.07}$

Table 1: Performance of GDRO Debiasing on top of our bias naming pipeline. Column *Unsup.* indicates if the method uses ground truth bias information. Best results for unsupervised debiasing methods are highlighted in bold. *No Val.Set in Bias-Id* highlights if the method does not rely on a validation set for inferring subgroups or bias-attributes (green tick, ✓) or it assumes having one (red cross, ✗). *Bias Named* indicates if the method extracts semantic names of found bias attributes. BAR lacks worst-group accuracy due to the absence of bias annotations. Average refers to the unbalanced test accuracy.

zero-shot classifier Radford et al. (2021) to infer the alignment of each sample towards the bias-keyword(s) of its target class or not. As a result, we obtain subgroup pseudo-labels, which can then be leveraged in a state-of-the-art supervised debiasing algorithm (e.g. GroupDRO Sagawa* et al. (2020)). For each dataset, we employ the following approach: given a training sample (x_i, y_i), and the set of keywords found for its class y_i, we use CLIP's image and text encoders to obtain its embeddings z_{img} and z_{txt}. A bias label is then computed for each sample by just annotating if the zero-shot classification from CLIP corresponds to a *bias-keyword* from its class or not. Finally, we plug our set of pseudo-labels in GroupDRO and measure the obtained test performances. In this stage, we employ the same hyperparameters used in the GroupDRO original implementation for CelebA and Waterbirds. For BAR, we set the learning rate and weight decay to 5×10^{-4} and 10^{-3} respectively, with a batch size of 128. Group adjustment is set to zero, and α is set to 0.5.

Results. Table 1 outlines the obtained results. Here, we categorized existing work according to two main factors: (i) Unsupervised (Unsup.), i.e. if the method does not use ground truth bias information (✓) or it does (✗); (ii) No Val.Set in Bias-Id, to better highlight whether bias information inference relies on the usage of a validation set with bias annotations. Our semantic bias discovery-based approach outperforms all the unsupervised methods not employing semantic information. To the only other work that mines bias-attributes in terms of text keywords (B2T+GDRO), we are still the best in terms of worst-group accuracy, while being in line concerning average accuracy (but with more stable results, as indicated by the lower standard deviation). Notably, both our method and B2T rely on GroupDRO for the final bias mitigation step, therefore our advantage has to be imputed on a finer semantic bias discovery. Additionally, our method does not rely on any validation set (e.g. BAR does not have one), and the amount of image captions we require is quite limited, thanks to the exemplar-mining step described in Sec. 3.2.1, whereas B2T directly captions the entire validation sets of each analyzed benchmark (For further comparisons with B2T, refer to Sec. G of the Suppl. Material). This feature allows our Bias-Discovery method to be employed in scenarios where bias annotations are not present at all, as in BAR.

5 Conclusion

In this work, we presented "Say My Name" (SaMyNa), a tool designed to identify and address biases within deep learning models semantically. Unlike similar methods that generate bias pseudo-labels without clear semantic information, SaMyNa offers a text-based pipeline that enhances the explainability of the bias extraction process from the model. Our approach, validated on well-known benchmarks, proved its effectiveness by both providing the keywords encoding the bias and assigning an interpretable score telling how much the model under analysis is biased to the found attribute. SaMyNa proposes itself not only as a post-hoc analysis tool: through its bias mining approach, it can determine the specific moment the model might be fitting a bias, for which there is in principle no need for a validation set to mine and name the bias. SaMyNa's ambition is to self-establish as a foundational tool in making deep learning models more transparent and fair, offering practical solutions for the scientific community and end-users alike.

REFERENCES

Mohsan Alvi, Andrew Zisserman, and Christoffer Nellåker. Turning a blind eye: Explicit removal of biases and variation from deep neural network embeddings. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 0–0, 2018.

Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. In *International Conference on Machine Learning (ICML)*, 2020a.

Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. In *International Conference on Machine Learning*, pp. 528–539. PMLR, 2020b.

Carlo Alberto Barbano, Benoit Dufumier, Enzo Tartaglione, Marco Grangetto, and Pietro Gori. Unbiased supervised contrastive learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=Ph5cJSfD2XN`.

Steven Bird, Edward Loper, and Ewan Klein. *Natural Language Processing with Python*. O'Reilly Media Inc., 2009.

Wieland Brendel and Matthias Bethge. Approximating cnns with bag-of-local-features models works surprisingly well on imagenet. *arXiv preprint arXiv:1904.00760*, 2019.

Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. Environment inference for invariant learning. In *International Conference on Machine Learning*, pp. 2189–2200. PMLR, 2021.

Moreno D'Incà, Elia Peruzzo, Massimiliano Mancini, Dejia Xu, Vidit Goel, Xingqian Xu, Zhangyang Wang, Humphrey Shi, and Nicu Sebe. Openbias: Open-set bias detection in text-to-image generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12225–12235, 2024.

Sabri Eyuboglu, Maya Varma, Khaled Saab, Jean-Benoit Delbrouck, Christopher Lee-Messer, Jared Dunnmon, James Zou, and Christopher Ré. Domino: Discovering systematic errors with cross-modal embeddings. *arXiv preprint arXiv:2203.14960*, 2022.

General Data Protection Regulation GDPR. General data protection regulation. *Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC*, 2016.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15262–15271, 2021.

Youngkyu Hong and Eunho Yang. Unbiased classification through bias-contrastive and bias-balanced learning. *Advances in Neural Information Processing Systems*, 34:26449–26461, 2021.

Pavel Izmailov, Polina Kirichenko, Nate Gruver, and Andrew G Wilson. On feature learning in the presence of spurious correlations. *Advances in Neural Information Processing Systems*, 35: 38516–38532, 2022.

Xu Ji, Joao F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9865–9874, 2019.

Eungyeup Kim, Jihyeon Lee, and Jaegul Choo. Biaswap: Removing dataset bias with bias-tailored swapping augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14992–15001, 2021.

Nayeong Kim, SEHYUN HWANG, Sungsoo Ahn, Jaesik Park, and Suha Kwak. Learning debiased classifier with biased committee. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 18403–18415. Curran Associates, Inc., 2022a. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/75004615747 1c56235a781f2eff6e226-Paper-Conference.pdf.

Nayeong Kim, Sehyun Hwang, Sungsoo Ahn, Jaesik Park, and Suha Kwak. Learning debiased classifier with biased committee. *Advances in Neural Information Processing Systems*, 35:18403–18415, 2022b.

Younghyun Kim, Sangwoo Mo, Minkyu Kim, Kyungmin Lee, Jaeho Lee, and Jinwoo Shin. Discovering and mitigating visual biases through keyword explanation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11082–11092, 2024.

Zhiheng Li, Anthony Hoogs, and Chenliang Xu. Discover and mitigate unknown biases with debiasing alternate networks. In *European Conference on Computer Vision*, pp. 270–288. Springer, 2022.

Daquan Liu, Chengjiang Long, Hongpan Zhang, Hanning Yu, Xinzhi Dong, and Chunxia Xiao. Arshadowgan: Shadow generative adversarial network for augmented reality in single light scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8139–8148, 2020.

Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 6781–6792. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/liu21f.html.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024. URL https://llava-vl.github.io/blog/2024-01-30-llava-next/.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Tambiama Madiega. Artificial intelligence act. *European Parliament: European Parliamentary Research Service*, 2021.

Tomáš Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pp. 746–751, 2013.

Yifei Ming, Hang Yin, and Yixuan Li. On the impact of spurious correlation for out-of-distribution detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 10051–10059, 2022.

Ron Mokady, Amir Hertz, and Amit H. Bermano. Clipcap: Clip prefix for image captioning, 2021. URL https://arxiv.org/abs/2111.09734.

Rémi Nahon, Van-Tam Nguyen, and Enzo Tartaglione. Mining bias-target alignment from voronoi cells. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4946–4955, 2023.

Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. *Advances in Neural Information Processing Systems*, 33:20673–20684, 2020.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. URL http://arxiv.org/abs/1908.10084.

Lorenzo Rosasco, Ernesto De Vito, Andrea Caponnetto, Michele Piana, and Alessandro Verri. Are loss functions all the same? *Neural computation*, 16(5):1063–1076, 2004.

Shiori Sagawa*, Pang Wei Koh*, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryxGuJrFvS.

Nimit Sohoni, Jared Dunnmon, Geoffrey Angus, Albert Gu, and Christopher Ré. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *Advances in Neural Information Processing Systems*, 33:19339–19352, 2020.

Enzo Tartaglione, Carlo Alberto Barbano, and Marco Grangetto. End: Entangling and disentangling deep representations for bias correction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13508–13517, 2021.

Enzo Tartaglione, Francesca Gennari, Victor Quétu, and Marco Grangetto. Disentangling private classes through regularization. *Neurocomputing*, 554:126612, 2023.

Haohan Wang, Zexue He, and Eric P. Xing. Learning robust representations by projecting superficial statistics out. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJEjjoR9K7.

Olivia Wiles, Isabela Albuquerque, and Sven Gowal. Discovering bugs in vision models using off-the-shelf image generation and captioning, 2023. URL https://arxiv.org/abs/2208.08831.

Qizhe Xie, Zihang Dai, Yulun Du, E. Hovy, and Graham Neubig. Controllable invariance through adversarial feature learning. In *NIPS*, 2017.

Xiangyi Yan, Shanlin Sun, Kun Han, Thanh-Tung Le, Haoyu Ma, Chenyu You, and Xiaohui Xie. After-sam: Adapting sam with axial fusion transformer for medical imaging segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 7975–7984, 2024.

Huaxiu Yao, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. Improving out-of-distribution robustness via selective augmentation. In *International Conference on Machine Learning*, pp. 25407–25437. PMLR, 2022.

Kunpeng Zhang, Liang Zhao, Chengxiang Dong, Lan Wu, and Liang Zheng. Ai-tp: Attention-based interaction-aware trajectory prediction for autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 8(1):73–83, 2022a.

Michael Zhang, Nimit S Sohoni, Hongyang R Zhang, Chelsea Finn, and Christopher Re. Correct-n-contrast: a contrastive approach for improving robustness to spurious correlations. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 26484–26516. PMLR, 17–23 Jul 2022b. URL https://proceedings.mlr.press/v162/zhang22z.html.

Yuhui Zhang, Jeff Z HaoChen, Shih-Cheng Huang, Kuan-Chieh Wang, James Zou, and Serena Yeung. Diagnosing and rectifying vision models using language. *arXiv preprint arXiv:2302.04269*, 2023.

Zhilu Zhang and Mert R. Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in Neural Information Processing Systems*, 2018-December:8778–8788, 5 2018. ISSN 10495258. doi: 10.48550/arxiv.1805.07836. URL https://arxiv.org/abs/1805.07836v4.

Supplementary Material

A Another case study for ImageNet-A: *nails* vs *mushrooms*

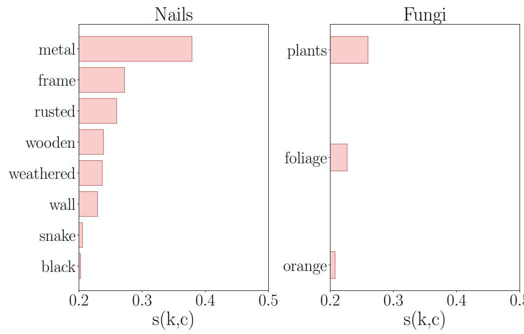


Figure 6: Similarity scores for the *nails* and *fungi* classes from ImageNet-A.

Besides the study provided in Sec. 4.3.2 of the main paper, we present here another case study on the ImageNet-A dataset, comparing two other critical classes: *nails* and *mushrooms*. Indeed, also for this case, the tested ResNet-50 model presents a big error between these two classes: is there a bias involved? Running our SaMyNa (Fig. 6), we observe that indeed there is a big correlation towards certain concepts for the *nails* class; however, these hardly resemble biases, but rather features of the target class. Indeed, we can easily imagine that concepts like `metal`, `frame`, or `rusted` can be easily associated with the target *nails* class. At this point, where is this big confusion arising from? The answer comes from a visual inspection of the samples, wherein multiple cases the shape factor of the two classes is extremely similar (both show a bulge on top and a thinner body underneath), making the classification task harder. In this case, we deduce that the model simply was unable to properly fit the two classes because of a lack of samples in the training set.

B Ablation Study

B.1 Ablations on the Bias Mining step

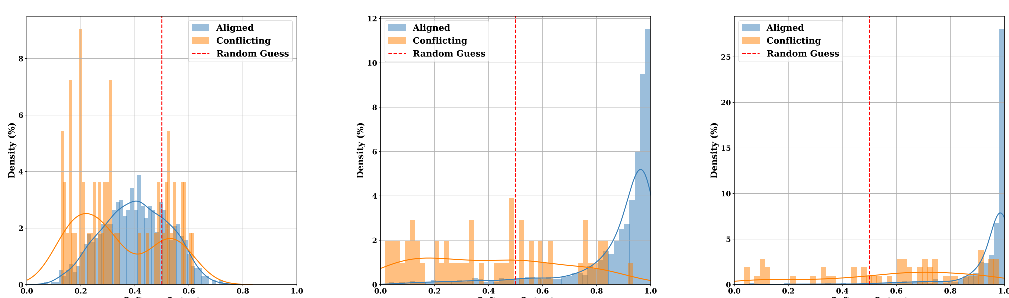


Figure 7: Output distributions *Waterbirds* target class from the ResNet-50 trained on Waterbirds at early stages (epoch 1, left), at extraction time t^* (epoch 6, center) and in the final stage (epoch 10, right).

In this section, we provide visualizations on the output distributions on the *Waterbirds* target when training a ResNet-50 on Waterbirds in the same setup as described in Sec. 4.1. Fig. 7 proposes visualizations of the output distributions for the bias-target aligned samples in blue (*waterbirds* and sea landscape), and bias-target conflicting samples in orange (waterbirds and ground landscape), in three different moments of the training: in the early stages (at $t = 1$, on the left, where t is the training epoch), at the chosen bias extraction time t^* ($t = 6$, at the center) and in the final stages ($t = 10$, on the right). We recall that the entire bias extraction process happens directly on the training set $\mathcal{D}^{\text{train}}$ and does not require the employment of a validation/test set. We observe here that, at extraction time, there is an evident separation between bias-aligned and bias-conflicting samples, and we are able to confidently isolate the most biased among the conflicting samples (the orange distribution having a larger population below the random guess threshold). This does not hold in case the extraction time is delayed, with the bias-conflicting distribution not exhibiting a peak anymore.

B.2 Ablation on Bias Naming

B.2.1 Ablation on Text Embedders

Embedding Model	Target	Keyword (value)
DistilRoberta	*Landbirds*	tree (0.26647)
	Waterbirds	marine (0.39470), ocean (0.36495), seagull (0.34526), coastal (0.34340), boat (0.33371), beach (0.33185), sea (0.31782), shoreline (0.30433), sandy (0.24345), waves (0.21502)
All-MiniLM-L12-v2	*Landbirds*	tree (0.34342), forest (0.31474), trees (0.30727), shrubs (0.28954), foliage (0.28264), vegetation (0.25404), branch (0.25149), leaves (0.21209)
	Waterbirds	sea (0.41955), ocean (0.37026), boat (0.35566), tide (0.34441), seagull (0.32581), flight (0.28337), waves (0.28000), coastal (0.27740), shoreline (0.25667), lake (0.23938), beach (0.23500), marine (0.22872)
MPNET	*Landbirds*	tree (0.38796), trees (0.32103), forest (0.30468), shrubs (0.25183), foliage (0.21930), branch (0.21547), bamboo (0.20619)
	Waterbirds	sea (0.39526), ocean (0.33487), beach (0.33385), shoreline (0.33176), waves (0.32562), lagoon (0.31923), coastal (0.30199), boat (0.26389), marine (0.24594), seagull (0.22533), water (0.20007)
NOMIC	*Landbirds*	small (0.24587), forest (0.21252), branch (0.20939)
	Waterbirds	sea (0.29738), ocean (0.28037), shoreline (0.27531), waves (0.27215), beach (0.25334), coastal (0.24178), tide (0.23727), seagull (0.22728), boat (0.22523), lagoon (0.21229), lake (0.20453), marine (0.20336)
AlBERT	*Landbirds*	tree (0.39055), branch (0.34739), trees (0.34374), forest (0.32672), shrubs (0.29123), vegetation (0.28895), foliage (0.28274), bamboo (0.25158), left (0.21710), leaves (0.21549)
	Waterbirds	beach (0.41926), ocean (0.39030), sea (0.38636), shoreline (0.35808), waves (0.35194), tide (0.34131), coastal (0.33961), marine (0.30102), sandy (0.28278), midflight (0.27492), boat (0.27448), seagull (0.25469), water (0.24652), lagoon (0.22010), flight (0.20572)

Table 2: Ablation study on Waterbirds, where we analyze the impact of employing a diverse encoder for SaMyNa.

In this subsection, we are interested in tasting SaMyNa with more diverse models for the textual embedding, in an attempt to check the generality of the proposed approach. We provide, in Tab. 2, the results obtained with five other popular textual embedding models. From our results, we can clearly see that when employing any of the tested models we are able to find back the two typical biases from waterbirds. We observe though that bigger encoders provide higher similarity scores (MPNET and AlBERT) , while smaller ones show lower scores, typically for the bias associated with *landbirds*. We can explain this to the higher complexity of capturing more diverse features associated with ground environments rather than maritime ones, requiring a higher capacity from the encoder. This pushes us to use, in the agnostic setup where we place ourselves, to use a large generic encoder.

B.2.2 ABLATION ON $\mathbf{f_{\min}}$

We propose here, in Tab. 3, the results we obtain for varying values for the hyperparameter $f_{\min}$, i.e. the minimum frequency with which a keyword has to appear so that it can be considered as a possible output. What is possible to observe is that the higher $f_{\min}$ the fewer and fewer keywords will be selected, and at some point, one class will be wrongfully marked as not holding a bias (with $f_{\min} = 0.60$). On the other hand, when not employing filtering at all ($f_{\min} = 0$), a lot of more fine-grained classes like `coniferous` or `brownish` arise. We find the chosen threshold ($f_{\min} = 0.15$) a fair compromise. As a sanity check, we also observe no change in the score for the remaining keywords.

$f_{\min} = 0.0$

Landbirds		*Waterbirds*	
tree	0.36	sea	0.53
shrubs	0.35	ocean	0.51
forest	0.34	beach	0.45
branch	0.33	waters	0.43
deciduous	0.33	shoreline	0.42
foliage	0.33	shore	0.41
branches	0.32	sailing	0.40
twigs	0.32	aquatic	0.40
trees	0.32	harbor	0.39
stalks	0.31	tide	0.39
forested	0.31	ship	0.38
plants	0.30	pier	0.38
plant	0.30	coastal	0.37
wooded	0.30	beachfront	0.37
leafy	0.29	coastline	0.36
woodland	0.27	water	0.33
vegetation	0.26	sailboat	0.32
brownishgray	0.24	marine	0.32
coniferous	0.24	boat	0.31
greenery	0.23	submerged	0.31
leafless	0.23	pond	0.30
brownishgrey	0.23	lagoon	0.29
leaves	0.22	waves	0.28
grasses	0.22	vessel	0.28
brownish	0.21	swimming	0.28
stalk	0.21	motorboat	0.27
		midflight	0.27
		boathouse	0.27
		lake	0.27
		wake	0.26
		wave	0.26
		flying	0.26
		boating	0.26
		dock	0.25
		river	0.25
		reef	0.25
		watermark	0.23
		cranes	0.23
		gliding	0.22
		fish	0.22
		seagulls	0.22
		hull	0.20
		flight	0.20

$f_{\min} = 0.1$

Landbirds		*Waterbirds*	
tree	0.36	sea	0.53
shrubs	0.35	ocean	0.51
forest	0.34	beach	0.45
branch	0.33	shoreline	0.42
foliage	0.33	shore	0.41
branches	0.32	tide	0.39
trees	0.32	coastal	0.37
stalks	0.31	water	0.33
forested	0.31	marine	0.32
plants	0.30	boat	0.31
wooded	0.30	submerged	0.31
vegetation	0.26	lagoon	0.29
coniferous	0.24	waves	0.28
greenery	0.23	midflight	0.27
leaves	0.22	lake	0.27
		river	0.25
		flight	0.20

$f_{\min} = 0.15$

Landbirds		*Waterbirds*	
tree	0.36	sea	0.53
shrubs	0.35	ocean	0.51
forest	0.34	beach	0.45
branch	0.33	shoreline	0.42
foliage	0.33	tide	0.39
trees	0.32	coastal	0.37
vegetation	0.26	water	0.33
greenery	0.23	marine	0.32
leaves	0.22	boat	0.31
		lagoon	0.29
		waves	0.28
		midflight	0.27
		lake	0.27
		flight	0.20

$f_{\min} = 0.30$

Landbirds		*Waterbirds*	
forest	0.34	ocean	0.51
branch	0.33	beach	0.45
trees	0.32	coastal	0.37
leaves	0.22	water	0.33
		waves	0.28
		midflight	0.27
		lake	0.27
		flight	0.20

$f_{\min} = 0.45$

Landbirds		*Waterbirds*	
branch	0.33	water	0.33
		waves	0.28
		flight	0.20

$f_{\min} = 0.60$

Landbirds		*Waterbirds*	
		water	0.33

Table 3: Ablation study on Waterbirds, where we analyze the impact of the threshold for the frequency for the found keywords ($f_{\min}$).

B.2.3 ABLATION ON k

We present here the ablation on k, that selects the cardinality of aligned and conflicting samples per class. Fig. 8 reports the study for 5 most occurring keywords in the cases under exam, while

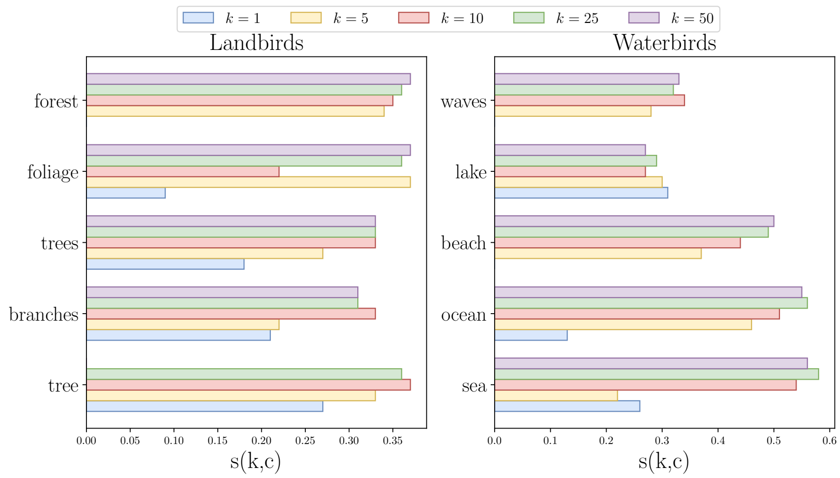


Figure 8: Ablation study on Waterbirds, where we analyze the impact of k that selects the cohort of images to extract keywords from.

the full results are later reported in Table 23. In the general case, we observe that for lower values of k the similarity score is in general lower, evidencing that the information extraction process is less accurate due to a general lack of information (and in general variety). This trend is particularly evident in more generic keywords like `forest`, `trees`, and `beach`. Finer-grained keywords like `foliage` and `waves` show a more irregular trend due to the specific sample selection. Overall we find that a fair compromise between performance and complexity is given by the intermediate $k = 10$ for which some keywords like `trees` reach a high value standing constant for higher values of k. We highlight that maintaining k at bay reduces the number of comparisons to perform, given that they grow quadratically.

B.2.4 Ablation on $\mathbf{t_{sim}}$

In this ablation study, we provide an example of the keywords resulting from SaMyNa when not employing any thresholding on the minimum similarity values for the found keywords. In Tab. 4, we provide the full output only for *landbirds*, due to the excessive length of the unfiltered output. However, this is not an issue because SaMyNa, when applied to binary classification, possesses a mathematical property that guarantees symmetrical keyword rankings for the two classes. Consequently, the keyword ranking for *waterbirds* is simply the inverse of the ranking for *landbirds*. Nevertheless, we provide the full list of the resulting keywords as a text file, available in the supplementary materials zip archive (`keywords_ablation_tsim.txt`). From an analysis of the emerging keywords, we observe three interesting intervals of values. High similarity values indicate those concepts that correlate well specifically with the learned class (filtered from the target class) and are those presented in the paper. Concepts whose similarity is close to zero are not correlated: indeed, we can find keywords like `long`, `muted`, and `given` that are neutral concepts. Interestingly, we can also identify concepts like `background` and `environment` that are super-classes of the two biases. This confirms that SaMyNa works properly since it puts itself in the best spot to best discriminate the two biases. Finally, the third region is for negatively correlated concepts (anti-correlated), where we easily find the concepts correlated with the other class (*waterbirds*) given that we are in a binary classification task.

B.2.5 Ablation on f_{min}, t_{sim}, and K

In this ablation study, we show the compound effect of the f_{min}, t_{sim}, and K hyperparameters. By setting $f_{min} = 0$, $t_{sim} = -1$ we effectively disable filtering. We also set $K = 50$ since it is the maximum value of K we tried, and will produce more captions, and, by extension, more keywords. Tab. 5 shows the results for both classes of Waterbirds. Due to space constraints we show only the top keywords. The total number of keywords ranked for each

Use of t_{sim}	Keyword (value)
$t_{sim} = -1$ (*Landbirds*)	tree (0.36349), shrubs (0.35333), forest (0.34249), branch (0.33068), foliage (0.32688), trees (0.31639), vegetation (0.25899), greenery (0.23251), leaves (0.22270), rounded (0.17975), bamboo (0.16404), green (0.16104), nature (0.14898), species (0.14680), brown (0.14534), small (0.13999), black (0.13964), gray (0.13468), natural (0.12627), habitat (0.12563), yellow (0.12454), bird (0.11301), patch (0.11042), perched (0.10776), soft (0.10717), neck (0.10514), lush (0.10440), slightly (0.09828), trunks (0.09826), standing (0.09586), color (0.09571), colors (0.09523), dark (0.09419), darker (0.09084), feet (0.09080), wildlife (0.08827), turned (0.08459), looking (0.07803), facing (0.07712), ecosystem (0.07514), palette (0.07354), lighter (0.07160), orange (0.07136), lighting (0.07126), position (0.06891), left (0.06699), markings (0.06692), positioned (0.06294), vibrant (0.05924), beak (0.05871), slender (0.05838), round (0.05730), side (0.05356), calm (0.05190), bright (0.04925), vertical (0.04921), pointed (0.04911), ground (0.04846), short (0.04838), red (0.04834), floor (0.04783), suggests (0.04622), blue (0.04544), tones (0.04530), passerine (0.04368), area (0.04230), plumage (0.04097), surrounding (0.04089), thin (0.03906), fallen (0.03622), gives (0.03523), birds (0.03506), main (0.03480), predominantly (0.03343), distinctive (0.03313), setting (0.03305), features (0.03300), giving (0.03169), found (0.03098), elements (0.03031), light (0.02905), type (0.02880), structures (0.02876), tall (0.02846), covered (0.02709), tail (0.02559), open (0.02540), feathers (0.02521), eyes (0.02402), appears (0.02250), suggesting (0.02136), indicating (0.02100), consists (0.02100), scattered (0.01934), wingtips (0.01928), gently (0.01406), subject (0.01349), might (0.01296), reflects (0.01232), possibly (0.01221), peaceful (0.01160), tranquil (0.01087), thriving (0.01059), sense (0.00703), presence (0.00641), tranquility (0.00393), tropical (0.00369), characterized (0.00365), suggest (0.00195), muted (0.00093), back (0.00062), seems (-0.00001), legs (-0.00070), near (-0.00084), towards (-0.00113), background (-0.00193), effect (-0.00199), eye (-0.00500), point (-0.00510), likely (-0.00543), providing (-0.00555), bare (-0.00591), given (-0.01045), environment (-0.01051), deep (-0.01359), visible (-0.01432), right (-0.01460), either (-0.01469), moment (-0.01475), could (-0.01496), long (-0.01563), dense (-0.01603), across (-0.01712), large (-0.01920), fully (-0.02127), focal (-0.02175), appear (-0.02263), creating (-0.02425), shows (-0.02666), healthy (-0.02720), cloudy (-0.02783), prominent (-0.02938), various (-0.03056), conditions (-0.03063), camera (-0.03170), photo (-0.03228), viewer (-0.03262), head (-0.03315), buildings (-0.03439), composition (-0.03512), overall (-0.03523), day (-0.03579), depicts (-0.03646), midst (-0.03986), daytime (-0.04005), focus (-0.04202), along (-0.04251), typical (-0.04274), taking (-0.04317), sunny (-0.04389), drawing (-0.04575), blurred (-0.04678), foreground (-0.04908), activity (-0.04970), contrast (-0.05024), wings (-0.05129), landscape (-0.05249), surroundings (-0.05353), packed (-0.05497), white (-0.05643), beauty (-0.05706), movement (-0.05968), weather (-0.06107), image (-0.06449), one (-0.06574), attention (-0.06792), captured (-0.06949), taken (-0.06966), extended (-0.06995), scene (-0.07093), view (-0.07824), clearly (-0.08422), atmosphere (-0.08640), mix (-0.08732), food (-0.08997), body (-0.09025), serene (-0.09172), similar (-0.09231), distance (-0.09281), backdrop (-0.09946), clouds (-0.10134), dynamic (-0.10504), surface (-0.10775), captures (-0.11025), overcast (-0.11541), world (-0.12448), sky (-0.12458), picturesque (-0.12616), horizon (-0.14756), clear (-0.15543), sandy (-0.15546), landing (-0.17462), seagull (-0.19202), flight (-0.20084), lake (-0.26879), midflight (-0.27087), waves (-0.28129), lagoon (-0.28892), boat (-0.31066), marine (-0.31538), water (-0.32785), coastal (-0.37410), tide (-0.38738), shoreline (-0.41605), beach (-0.45110), ocean (-0.51175), sea (-0.52685)

Table 4: Ablation study on Waterbirds, where we analyze the impact of the threshold for the similarity score (t_{sim}) for the class *landbirds*.

class is 1547 and the full result can be consulted in the supplementary materials zip archive (`keywords_ablation_all_hyperparams.txt`).

C SaMyNa on an unbiased dataset

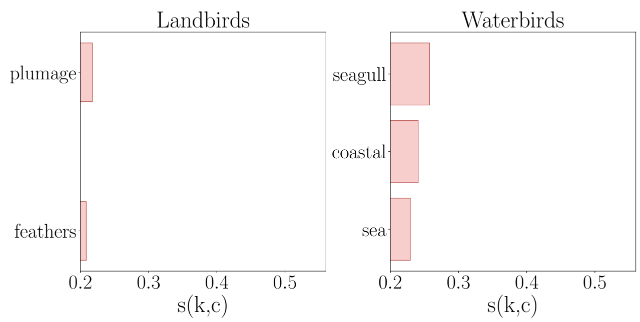


Figure 9: Ablation study on Waterbirds, where we balance the two classes, a-priori removing the bias.

We propose here a study on a virtually balanced version of the Waterbirds dataset. Fig. 9 reports the results in a graphical form, while Tab. 22 in a later section reports the numerical values. While we should have a-priori removed the bias by balancing the dataset, resulting in a general, massive reduction of the similarity scores, we still observe some mild correlations arising, especially for the *waterbirds* class. Specifically, besides the `seagull` keyword evidencing a (potential) higher presence of seagulls in the data split, we still see some concepts like `coastal` and `sea` correlating with the learned class. This is expected: given that these features are easy to learn, the model still captures them, but the low similarity score indicates that it does not heavily rely on them. This shows that, despite balancing the dataset, some biased features can still permeate through the model, depending on how easy they are to capture. This further motivates our work, focusing on model debiasing rather than dataset debiasing. The presence of `plumage` of `feathers` keywords for *landbirds* indicates a potential feature for this specific class, and the very low correlation does not pose a big bias threat.

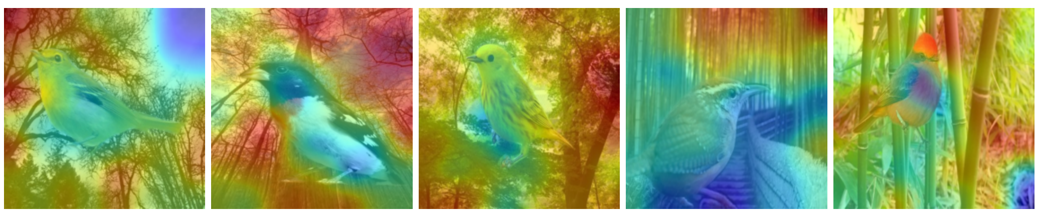
D Bias discovery on Vision Transformer models

We present here a study on two popular pre-trained Vision Transformers architectures: ViTb-16 and Swin-V2. Tab. 6 reports the outcome of SaMyNa for the classes *crayfish*, *rhinoceros beetle*, *stick insect* and *cockroach* of ImageNet-A. Despite the potential of generalization for these architectures, we are still able to observe, although with different magnitudes, some biases. Regarding ViTb-16, the class *crayfish* is still associated with `meal` and *cockroach* is associated with `floor`: interestingly the impact of `hand` for the *stick insect* is heavily reduced compared to the ResNet model, while with the introduction of sliding windows it goes back up for Swin-V2. In general, we notice that these architectures, although still suffering from bias, are less prone to it, probably due to finer training enhanced by larger parametrization combined with the self-attention mechanism they embody.

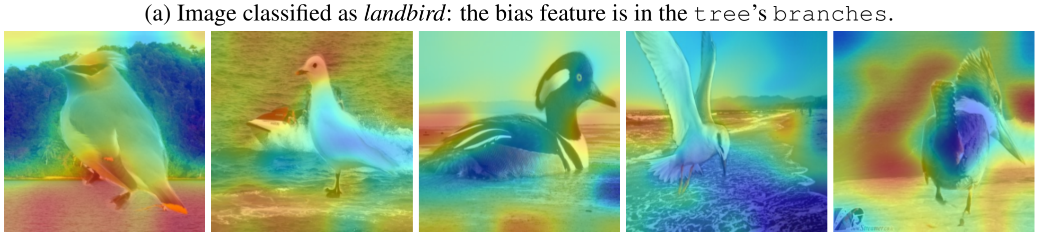
E Visual feedback

For visualization purposes only, SaMyNa can leverage a visual encoder instead of a text encoder to identify the part of the image where the potential bias is located. To do this, we adopt the same strategy described in Sec. 3.2.4 to generate the learned class embeddings $E^*(c)$ directly using image embeddings generated with CLIP's visual encoder Radford et al. (2021)[4]. We can then compare $E^*(c)$ with the embeddings of patches from the image we want to analyze. Fig. 10a and Fig. 10b highlight (in red) that the most salient feature is the tree for the landbird, while it is the sea for the waterbird: the model under exam does not focus on the birds but rather on the background, coherently with what we have observed in Sec. 4.3.1.

[4]`https://huggingface.co/sentence-transformers/clip-ViT-L-14`



(a) Image classified as *landbird*: the bias feature is in the `tree`'s `branches`.



(b) Image classified as *waterbird*: the bias feature is the `sea`.

Figure 10: Bias heatmaps generated with SaMyNa using CLIP's vision encoder Radford et al. (2021) on Waterbirds.

Class	Keyword (value)
Landbirds	forest (0.37326), foliage (0.37124), shrubs (0.36617), deciduous (0.35358), tree (0.35058), forested (0.34944), twigs (0.34178), stalks (0.33679), wooded (0.33528), plants (0.33146), leaf (0.33143), woodland (0.32835), plant (0.32778), trees (0.32101), branch (0.31242), leafy (0.30757), vegetation (0.30122), fern (0.29482), ferns (0.29417), pine (0.29382), branches (0.29285), jungle (0.28948), woodpecker (0.28427), evergreens (0.28349), lilies (0.26924), evergreen (0.26723), grasses (0.26579), flora (0.26248), garden (0.26248), brownishblack (0.25752), brownishgray (0.25395), coniferous (0.25326), twig (0.25266), leaves (0.24210), brownishgrey (0.24178), leafless (0.23821), lichen (0.23721), cultivated (0.23228), meadow (0.23000), undergrowth (0.22943), driftwood (0.22929), brownish (0.22868), greenery (0.22802), greenishblack (0.22562), stalk (0.22101), bark (0.21466), grove (0.21376), greyish (0.20718), blackbird (0.20627), wood (0.19627), redwoods (0.19363), wildflower (0.19303), greenishbrown (0.19213), moss (0.19205), trunk (0.19026), grayishbrown (0.18905), greenishyellow (0.18752), seeds (0.18637), bamboolike (0.17961), stripe (0.17889), stripes (0.17887), squirrel (0.17485), reddishbrown (0.17260), rounded (0.17222), yellowishgreen (0.17070), green (0.16889), hunting (0.16734), trail (0.16718), grass (0.16713), greenish (0.16563), stem (0.16458), grassland (0.16286), brown (0.16143), bamboo (0.16072), habitat (0.15888), fence (0.15749), grayishblack (0.15692), wooden (0.15679), bloom (0.15548), species (0.15385), broadleafed (0.15363), yellowishbrown (0.15172), rural (0.15167), lining (0.15158), foraging (0.15140), greenishblue (0.15121), shaded (0.15096), yellowish (0.14973), grayish (0.14877), crouching (0.14802), striped (0.14795), pouch (0.14582), ruffled (0.14575), songbird (0.14521), songbirds (0.14500), biodiversity (0.14497), agile (0.14429), ears (0.14411), litter (0.14373), nature (0.14291), cracks (0.14155), nesting (0.13882), flowers (0.13817)
Waterbirds	sea (0.57176), ocean (0.56346), seas (0.54325), beach (0.50411), seaside (0.49274), watercraft (0.45897), waters (0.45848), seascape (0.45239), shoreline (0.44961), shore (0.44412), tide (0.44050), coast (0.43822), coastal (0.42281), maritime (0.41680), aquatic (0.41262), pier (0.40740), sailing (0.40685), beachfront (0.40225), harbor (0.39950), coastline (0.39708), ships (0.37315), marina (0.37144), water (0.36510), ship (0.35923), waves (0.34343), waterfront (0.34084), marine (0.32861), sails (0.31933), boats (0.31620), sailboat (0.31366), wave (0.31221), floating (0.30932), lagoon (0.30155), submerged (0.29920), swim (0.29753), bay (0.29548), wet (0.29291), pond (0.29052), whale (0.28699), swimming (0.28541), surfboard (0.28261), wake (0.27740), boat (0.27665), midflight (0.27397), cliffs (0.27336), fish (0.27117), dock (0.27024), reef (0.26725), glide (0.26585), lake (0.26331), vessel (0.25468), sand (0.25399), river (0.25284), flying (0.25184), seagulls (0.24805), boathouse (0.24752), pacific (0.24669), cranes (0.24276), watermark (0.24022), surfing (0.23908), fluid (0.23850), pool (0.23368), docked (0.22785), motorboat (0.22620), seagull (0.22198), boating (0.21686), soaring (0.21171), hull (0.21147), coral (0.21051), island (0.20324), strait (0.20151), skyline (0.19948), sandy (0.19563), gliding (0.19526), paddleboarding (0.19393), jetty (0.18614), float (0.18407), lakeside (0.18301), islands (0.17514), seabirds (0.17263), flight (0.17119), midair (0.17063), landing (0.17029), clear (0.16458), dive (0.16369), horizon (0.16368), waterfowl (0.15734), docking (0.15089), extreme (0.15039), seabird (0.14445), screensaver (0.14153), ripples (0.14057), picturesque (0.13896), whimsy (0.13849), damp (0.13590), inflated (0.13585), floats (0.13464), motion (0.13384), immersed (0.13378), flowing (0.13237), mirrorlike (0.13182), cityscape (0.13109), fly (0.13057), air (0.13029), whitecapped (0.12935), balloon (0.12932), overcast (0.12896), soars (0.12856), world (0.12841)

Table 5: Ablation study on Waterbirds, where we analyze simultaneously the impact of t_{sim}, f_{min}, and K. Filtering is disabled by setting $t_{sim} = -1$ and $f_{min} = 0$. We use $K = 50$ to generate more captions and, as a consequence, more keywords.

Tested architecture	Target	Keyword (value)
ViTb-16	*Crayfish*	crab (0.41234), meal (0.36249), crustacean (0.27688), food (0.26259), plate (0.20828)
	Rhinoceros Beetle	insect (0.26927), tree (0.26825),
	Stick Insect	plant (0.43678), garden (0.34558), leaves (0.32101), greenery (0.26056), tree (0.23140), thumb (0.23000), hand (0.22839), green (0.22577), grasshopper (0.22324), cricket (0.21173),
	Cockroach	floor (0.33605), dark (0.32662), debris (0.28657), darker (0.26946), lighting (0.25669), black (0.24747), markings (0.23064), color (0.21767), surface (0.21696), colors (0.21068), insect (0.20682)
Swin-V2 B	*Crayfish*	lobster (0.56152), crab (0.42682), aquatic (0.34159), water (0.23999)
	Rhinoceros Beetle	darkcolored (0.25085), greenery (0.24698), park (0.24086), black (0.23795), colors (0.20523), color (0.20430), nature (0.20135)
	Stick Insect	plant (0.32613), hand (0.32460), finger (0.27872), garden (0.26617), leaves (0.25800), greenery (0.21344)
	Cockroach	floor (0.28899), shadows (0.22149), insect (0.21984), debris (0.20011)

Table 6: Testing pre-trained Vision Transformer architectures on *crayfish*, *rhinoceros beetle*, *stick insect* and *cockroach* classes from ImageNet-A.

Keywords	**ClipCap**	**LLaVA 34b**
old/oldest	0.04%	2.00%
middle-age/middle-aged	0.00%	4.50%
young/youngest	0.86%	10.0%
blond/blonde	0.02%	49.0%
smile/smiling/smiles	1.89%	58.0%
tie	0.05%	2.00%
eyeglasses/glasses/sunglasses	0.69%	3.00%
beard	1.70%	4.50%
mustache	0.25%	3.50%
makeup	1.27%	61.0%
man/male	9.26%	22.0%
woman/female	2.21%	71.5%
hat/hats	0.84%	2.00%
earring/earrings	0.02%	14.5%
necklace	0.02%	8.50%

Table 7: The table shows the percentage of captions that contain keywords related to CelebA attributes, both for ClipCap and LLaVA-34B. As can be seen, LLaVA-34B is able to capture much more attributes from CelebA's images. Thus, ClipCap is not suitable for detecting biases other than "man" on CelebA.

F Comparison of different captioners

We perform a comparison between different captioners, notably with ClipCap (Mokady et al., 2021) as it has been recently used by related works (Kim et al., 2024). First of all, we are interested in evaluating whether ClipCap can be used to detect biases on CelebA, and how it compares to LLaVA-34b. For this, we report in Tab. 7 the percentage of captions that contains keywords related to CelebA's attributes. As can be seen, ClipCap detects these concepts very rarely, except for "man", which is detected in 10% of the images. LLaVA-34b, on the other hand, detects these keywords much more frequently, for example it detects "man" in 22% of the images, while "woman" is detected in 71.5% of the images (totaling 93.5% detection for gender related keyword, compared to ClipCap's 11.47%). This is a consequence of ClipCap's short captions, that cannot capture enough information from the images. ClipCap's caption are 9 words long on average, while LLaVA's are 137 words long on average. Finally, we show a qualitative comparison of ClipCap's and LaVA's caption in Tab. 8 and Tab. 9.

Image	ClipCap	LLaVA-34B
	actor with a weight of kg and a feet size of 9 in favorite outfit & clothing style.	The image is a close-up portrait of a woman with fair skin and light-colored hair. Her hair is styled in a side-swept manner, with some strands falling over her forehead. She has a neutral expression on her face and is looking directly at the camera. The woman is wearing makeup, including eyeliner and lipstick, which accentuates her features. She has a small earring visible in her left ear. The background of the image is plain and does not provide any additional context about the location or setting. The lighting on the subject is soft and even, highlighting her facial features without creating harsh shadows. The style of the image suggests it could be from a red carpet event or a formal photoshoot, given the subject's attire and the professional quality of the photograph.
	actor was a popular actor and comedian.	The image is a portrait-style photograph of a man. He appears to be middle-aged with short, dark hair and a fair complexion. The man is wearing a patterned shirt with horizontal stripes in various colors, including red, white, and black. He has a slight smile and is looking directly at the camera. The background is a solid, light blue color, providing a neutral backdrop that contrasts with the subject. There are no visible texts or distinguishing marks on the image that provide additional context or information about the location or the identity of the man. The style of the image is straightforward and seems to be a standard headshot or portrait commonly used for identification or promotional purposes.

Table 8: Comparison of captions generated by ClipCap and LLaVA-34B on CelebA. The images to caption were selected randomly.

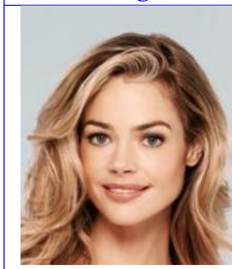
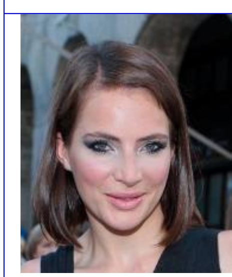
Image	ClipCap	LLaVA-34B
	actor is the face of beauty.	The image is a close-up portrait of a woman with a soft, natural appearance. She has long, wavy blonde hair that falls over her shoulders. Her facial features are symmetrical, with a straight nose, full lips, and clear skin. Her eyes are a light color, possibly blue or green, and she is looking directly at the camera with a gentle expression. The woman's makeup is subtle, enhancing her natural beauty without appearing overly dramatic. The background of the image is a solid, neutral color, providing a clean and uncluttered backdrop that allows the focus to remain on the woman. The lighting in the photograph is soft and even, which contributes to the gentle and approachable look of the subject. There are no visible texts or logos in the image, and the style of the photograph suggests it could be used for a variety of purposes, such as a professional headshot, a beauty advertisement, or a personal portrait. The image does not provide any additional context about the location or setting, as the focus is solely on the woman's face and upper body.
	actor with her hair in a bob.	The image is a close-up portrait of a woman with a smiling expression. She has short, dark hair and is wearing makeup that includes dark eyeliner and eyeshadow, as well as what appears to be a light foundation or powder on her face. Her lips are slightly parted, and she is looking directly at the camera. The woman is wearing a black garment that is not fully visible in the frame. The background is blurred, but it suggests an indoor setting with architectural features such as arches and what might be a stone or brick wall. The lighting on the subject is bright, highlighting her features and the contours of her face. The style of the image is a standard portrait with a focus on the subject's face and expression. There are no visible texts or logos in the image.

Table 9: Comparison of captions generated by ClipCap and LLaVA-34B on CelebA. The images to caption were selected randomly.

Method	Captioning Time (LLAVA 34B)
SaMyNa (K=1)	17 minutes
SaMyNa (K=5)	86 minutes
SaMyNa (K=10)	3 hours
SaMyNa (K=25)	7 hours
SaMyNa (K=50)	14 hours
B2T	60 days

Table 10: Comparison of captioning runtime between SaMyNa (for different values of K) and B2T using LLaVA-34B on CelebA. B2T must caption the whole validation set of CelebA, which is composed of about 19k images, while SaMyNa uses bias-mining to select a sample of $K * C * 2$ images, where C is the number of classes. By default, SaMyNa uses $K = 10$, for a total of 40 samples on CelebA.

G Comparison with B2T

In this section, we provide a more in-depth comparison with the B2T algorithm proposed by Kim et al. (2024). B2T is relevant to our work, as it shares the same goal of extracting human-interpretable descriptions of potential biases affecting visual models. The key differences between SayMyNa and B2T can be summarized as follows:

- SayMyNa does not require a validation set for bias discovery, in contrast to B2T;
- SayMaNa can extract few candidate exemplars directly from the training set thanks to the Bias Mining step.

This represents a key aspect in unsupervised bias discovery and mitigation, as in a realistic scenario a validation set comprising conflicting samples is rarely available (like in BAR or BFFHQ). Besides, B2T requires captioning the entire validation set in order to extract relevant biases from conflicting samples. In contrast, SayMyNa is much more efficient and allows for the usage of larger and more accurate captioners such as LLaVa-34B (see Sec. F for a comparison between ClipCap and LLaVa-34B).

Why B2T cannot leverage better captioners while SayMyNa can The quality of extracted keywords directly depends on the quality of the captioner. B2T is forced to employ smaller and quicker captioners such as ClipCap, which, however, provides less accurate captions when compared to models such as LLaVa-34B. Tab 10 shows the time required for SaMyNa and B2T on the CelebA dataset using LLaVa-34B on an NVIDIA A40 equipped with 48 GB, tested on batch size 5.

Why B2T requires a full validation set and SayMyNa does not To showcase of B2T requires a large enough validation set in order to extract accurate keywords, we compare the keywords extracted by SayMaNa and B2T with varying sample size. The results are presented in Tab. 11. We highlight cases in which the relevant keyword was ranked higher than the other method. The results clearly show that our method, which leverages the bias mining step, is consistently more accurate than B2T in extracting the right keywords. Furthermore, keep in mind that B2T keywords need to be inverted as reported in the original paper (e.g. woman → man), thus for K=1, B2T actually predicts the opposite bias. This is straightforward for a binary attribute such as CelebA's gender, but not obvious when more than two biases are present in the training set. For completeness of results, we report the full ranking of both algorithms for all values of K, in Tab. 12 (K=1), Tab. 13 (K=5), Tab. 14 (K=10), Tab. 15, Tab. 15, and Tab. 16.

Difference in keyword filtering An important novelty of SayMyNa is that it can find an embedding vector that represents the bias of the model in a certain class. This embedding vector is found by doing arithmetic operations between the embeddings of the captions as explained in the paper. This means that we solve the problem of synonyms. In B2T there is a heavy filtering step before the ranking of the keywords that uses the YAKE keyword extraction algorithm. YAKE does not take into account the semantics of words, which means that it may filter out synonyms if they do not reach

Class	**Method**	**Expected**	**K=1**	**K=5**	**K=10**	**K=25**	**K=50**
Blond	**B2T**	**man**	woman (6th)	N/A	N/A	N/A	N/A
	SaMyNa	**woman**	N/A	woman (6th)	woman (1st)	woman (5th)	woman (1st)
Not Blond	**B2T**	**woman**	N/A	woman (5th)	woman (3rd)	woman (5th)	woman (4th)
	SaMyNa	**man**	male (1st)	male (1st)	man (1st)	man (1st)	man (1st)

Table 11: Comparison between SaMyNa and B2T using the same captioner (LLaVA-34B) and using an equally sized subsample of CelebA's validation set. The number of sample images is $K * 4$. For B2T we selected K random images for the correctly classified, and K for the incorrectly classified examples of each class. For SaMyNa we use our subsampling algorithm. B2T's expected answer is the opposite of SaMyNa's. We show the position in the ranking of gender keywords. For both algorithms, the default filtering method is used, except for SaMyNa, where we don't filter the target class since it's also not filtered by B2T. As can be seen, SaMyNa detects the bias for "not blond" every time, while for the "not blond" class fails only for $K = 1$. B2T on the other hand detects the bias for the "not blond" class 4 times out of 5 with worse ranking positions than SaMyNa, while for the "blond" class it never detects the bias, and for $K = 1$ it's answer is the opposite of the expected answer.

Blond (B2T)		Blond (SaMyNa)		Not blond (B2T)		Not blond (SaMyNa)	
Keyword	CLIP Score	Keyword	Cosine Similarity	Keyword	CLIP Score	Keyword	Cosine Similarity
wavy	1.844	earrings	0.27081	blonde hair	1.594	male	0.32069
fair complexion	1.781	makeup	0.26793			grass	0.24336
close-up	1.609	blonde	0.21713			subject	0.23409
close-up photograph	1.5625	lipstick	0.20272			environment	0.21606
neutral expression	1.375	eyeshadow	0.20149			camera	0.21347
woman	1.078					capturing	0.21132
complexion	0.7344					mood	0.20288
lips slightly	0.4688						
complexion and long	0.375						
lips slightly parted	0.375						
hair	0.2812						

Table 12: Full keyword rankings of B2T and SaMyNa for the results shown in Tab. 11 with $K = 1$. As can be seen, while SaMyNa fails for the "blond" class, it detects keywords that still make sense like "makeup", "earrings", and "lipstick", which are usually associated with woman. While B2T fails in a worse way, since it's answer is the polar opposite of the expected one (it should answer male). B2T also fails for the "not blond" class.

Blond (B2T)		Blond (SaMyNa)		Not blond (B2T)		Not blond (SaMyNa)	
Keyword	CLIP Score	Keyword	Cosine Similarity	Keyword	CLIP Score	Keyword	Cosine Similarity
provide additional context	0.5938	blonde	0.38934	wavy blonde hair	4.734	male	0.30272
provide	0.5625	makeup	0.34895	wavy blonde	3.922	man	0.27394
distinguishing	0.5	mascara	0.33196	blonde	3.719	shirt	0.20739
additional context	0.4844	lipstick	0.31817	blonde hair	3.64		
context	0.4688	eyeliner	0.31324	woman	1.844		
texts	0.3125	woman	0.29510	wearing makeup	1.3125		
distinguishing marks	0.2656	eyeshadow	0.24667	eyeliner	1.297		
additional	0.1562	shadows	0.24538	eyes	1.125		
provide additional	0.1406	hair	0.23935	makeup	1.078		
style	0.03125	face	0.22412	camera	0.9062		
		head	0.20731	expression	0.703		
		styled	0.20134	long	0.578		
				directly	0.5156		
				wearing	0.2188		
				hair	0.0		

Table 13: Full keyword rankings of B2T and SaMyNa for the results shown in Tab. 11 with $K = 5$.

Blond (B2T)		Blond (SaMyNa)		Not blond (B2T)		Not blond (SaMyNa)	
Keyword	CLIP Score	Keyword	Cosine Similarity	Keyword	CLIP Score	Keyword	Cosine Similarity
eyeliner	0.9688	woman	0.39947	wavy blonde hair	3.594	man	0.45820
wearing makeup	0.4844	blonde	0.34301	blonde hair	2.938	male	0.39175
photograph	0.03125	mascara	0.24959	woman	1.828		
		makeup	0.24877	light-colored hair	1.359		
		lipstick	0.22272	close-up	0.01563		
		eyeliner	0.20065				

Table 14: Full keyword rankings of B2T and SaMyNa for the results shown in Tab. 11 with $K = 10$.

Blond (B2T)		Blond (SaMyNa)		Not blond (B2T)		Not blond (SaMyNa)	
Keyword	CLIP Score	Keyword	Cosine Similarity	Keyword	CLIP Score	Keyword	Cosine Similarity
visible texts	1.0	blonde	0.45956	wavy blonde hair	3.766	man	0.40512
close-up photograph	0.7656	makeup	0.32295	blonde	3.547		
provide additional context	0.6406	mascara	0.32257	blonde hair	3.438		
directly	0.5938	lipstick	0.30535	blonde hair styled	3.438		
additional context	0.5156	woman	0.30441	woman	1.594		
portrait-style photograph	0.4062	eyeliner	0.24918	wearing makeup	0.797		
fair	0.4062	hair	0.22028	makeup	0.672		
face	0.1875	eyeshadow	0.20621	fair skin	0.3906		
				eyes	0.2812		
				hair styled	0.2344		
				styled	0.2344		
				portrait	0.2188		

Table 15: Full keyword rankings of B2T and SaMyNa for the results shown in Tab. 11 with $K = 25$.

Blond (B2T)		Blond (SaMyNa)		Not blond (B2T)		Not blond (SaMyNa)	
Keyword	CLIP Score	Keyword	Cosine Similarity	Keyword	CLIP Score	Keyword	Cosine Similarity
close-up photograph	0.8438	woman	0.42156	wavy blonde hair	4.0	man	0.47463
visible texts	0.797	blonde	0.39872	blonde hair	3.5	shirt	0.20972
photograph	0.6875	mascara	0.29177	blonde	3.406		
expression	0.5	lipstick	0.29163	woman	1.375		
additional context	0.4531	makeup	0.26443	wearing makeup	0.7188		
provide additional context	0.4375			makeup	0.5625		
visible	0.3594			close-up	0.1875		
person	0.2812			eyeliner	0.1719		
texts or distinguishing	0.2656			fair	0.1562		
provide additional	0.25			hair	0.0781		
distinguishing marks	0.1875						
style	0.1875						
wearing	0.1406						
face	0.03125						

Table 16: Full keyword rankings of B2T and SaMyNa for the results shown in Tab. 11 with $K = 50$.

a certain frequency threshold individually. Conversely, our method does a very lightweight filtering of keywords before ranking, removing only very rare keywords that may result from captioning mistakes. After this lightweight filtering, we work entirely in the embedding space of the text embedder, which can account for all the synonyms and construct an embedding vector that represents the bias itself semantically. Keyword embeddings are then compared to the bias embeddings and ranked according to cosine similarity, our heavy filtering is done after ranking by setting a similarity threshold. Evidence of B2T's filtering being too heavy is found in the full keyword rankings for the class "not blond" of CelebA in the supplementary material of B2T, where the keyword "woman" does not survive filtering and does not appear in the ranking. In particular, B2T selects the top 20 best keywords according to YAKE before ranking, while we discard keywords that do not appear in at least 15.

Symmetraical keywords ranking Additionally, our algorithm has the interesting mathematical property that for binary classification datasets, the ranking for one bias class is symmetrical with respect to the other class (because the two bias embedding vectors point in opposite directions, so the cosine similarity will give opposite scores).

SayMyNa can be applied on images In Sec. E, we show the possibility of SayMyNa to work on other modalities without involving text. We use image embeddings instead of caption embeddings to produce the embedding vectors that represent the biases, and then we rank image patches instead of keywords according to the similarity of the patch to the bias embedding and we display this as a heatmap. This is a further novelty of SayMyNa with respect to B2T, showing that the underlying mechanism is fundamentally different (B2T only works with CLIP-like models and needs both images and text).

H OUTPUTS OF SAMYNA

In this section, we provide the detailed output of our bias naming process for the experiments described in the main paper. We report the obtained keywords alongside their associated cosine similarity, in decreasing order of value, for Waterbirds (Tab. 17), CelebA (Tab. 18), BAR (Tab. 19), ImageNet-A (Tab. 20 and 21), on a completely balanced version of Waterbirds (Tab. 22) and for the ablation study on k (Tab. 23).

Landbirds		*Waterbirds*	
Keyword	Cosine Similarity	Keyword	Cosine Similarity
tree	0.37119	sea	0.55190
forest	0.36188	ocean	0.53744
trees	0.35160	beach	0.43492
forested	0.35120	waters	0.41526
foliage	0.34351	shoreline	0.39085
branch	0.31807	shore	0.37826
stalks	0.31405	coastal	0.37666
vegetation	0.26898	water	0.31551
leaves	0.23139	boat	0.29067
		midflight	0.28737
		waves	0.27717
		seagull	0.26215
		lagoon	0.24614
		flight	0.22423
		lake	0.20760

Table 17: Similarity scores for the Waterbirds dataset.

Not blonde		*Blonde*	
Keyword	Cosine Similarity	Keyword	Cosine Similarity
man	0.40964	woman	0.48667
male	0.38960	makeup	0.25876
		lipstick	0.23692
		eyeshadow	0.20672

Table 18: Similarity scores for the CelebA dataset.

Climbing		Diving		*Fishing*		*Racing*		*Throwing*		*Vaulting*	
Keyword	Cosine Similarity	Keyword	Cosine Similarity	Keyword	Cosine Similarity	Keyword	Cosine Similarity	Keyword	Cosine Similarity	Keyword	Cosine Similarity
cliff	0.52706	scuba	0.57237	boat	0.45474	cars	0.37512	pitch	0.55673	midair	0.41947
rock	0.44902	underwater	0.54212	river	0.45005	car	0.35930	baseball	0.52524	jump	0.40536
rocks	0.40822	submerged	0.38070	sea	0.42662	track	0.32670	pitcher	0.50494	pole	0.33902
steep	0.35067	coral	0.28247	ocean	0.38038	stadium	0.26144	batter	0.48165	high	0.31889
backpack	0.32901	depths	0.27969	lake	0.35662	speeds	0.24373	player	0.47083	suspended	0.31659
rocky	0.32047	ocean	0.25977	water	0.34019	batter	0.23747	mound	0.42858	athleticism	0.31222
rugged	0.25324	sea	0.25319	marine	0.29564	competitive	0.23413	glove	0.41766	bar	0.27459
jagged	0.24785	wetsuit	0.24872	waves	0.27669	speed	0.22976	athlete	0.40207	outstretched	0.25279
adventurous	0.21917	swimming	0.24449	coral	0.24126	competition	0.22619	playing	0.39830	agility	0.24972
trail	0.21789	marine	0.23934	underwater	0.23954	asphalt	0.22051	sports	0.39558	upward	0.24567
strength	0.21506	water	0.22909	wetsuit	0.21871	baseball	0.21841	sport	0.38070	athletic	0.23762
exploring	0.21200			rocks	0.21742	pitch	0.21645	field	0.36654	athlete	0.22750
nature	0.21199			divers	0.21735	team	0.20312	elbow	0.36480	casting	0.20631
ascending	0.21090			rocky	0.20966	uniform	0.20134	ball	0.35208	prowess	0.20590
adventure	0.20970			scuba	0.20206	pitcher	0.20080	athletic	0.35054	highstakes	0.20370
expedition	0.20443							game	0.34915	arched	0.20291
								team	0.31384	sky	0.20140
								athleticism	0.30591		
								arm	0.28327		
								stadium	0.27959		
								jersey	0.26615		
								main	0.25232		
								action	0.25062		
								arms	0.24658		
								activity	0.24016		
								striking	0.23857		
								outstretched	0.23707		
								position	0.23591		
								uniform	0.23211		
								serving	0.22028		
								actions	0.21900		
								focused	0.21649		
								skill	0.21620		
								foreground	0.21460		
								center	0.21259		
								stands	0.21227		
								emphasizes	0.20999		
								physical	0.20860		
								catch	0.20644		
								emphasizing	0.20625		
								overall	0.20215		
								agility	0.20136		

Table 19: Similarity scores for the BAR dataset.

Crayfish		*Rhinoceros Beetle*	
Keyword	Cosine Similarity	Keyword	Cosine Similarity
crab	0.39954	forest	0.27621
meal	0.31551	black	0.24731
crustacean	0.28440	branch	0.20823
food	0.23936	stands	0.20675
sandy	0.20224		
Stick Insect		*Cockroach*	
Keyword	Cosine Similarity	Keyword	Cosine Similarity
hand	0.41610	insect	0.29116
thumb	0.39488	creature	0.27834
finger	0.36924	darkcolored	0.27402
touch	0.33881	beetle	0.26163
plant	0.31364	dark	0.24713
plants	0.30637	flattened	0.24523
foliage	0.28583	grasshopper	0.24468
garden	0.27045	crustacean	0.22505
field	0.22898	darker	0.22500
leaves	0.21886	floor	0.21054
forest	0.21871	black	0.20142
holding	0.21059		
grasshopper	0.20454		

Table 20: Similarity scores for the *crayfish*, *rhinoceros beetle*, *stick insect* and *cockroach* classes from ImageNet-A.

Nails		*Mushrooms*	
Keyword	Cosine Similarity	Keyword	Cosine Similarity
metal	0.37880	plants	0.25968
frame	0.27216	foliage	0.22674
rusted	0.25972	orange	0.20766
wooden	0.23891		
weathered	0.23700		
wall	0.22942		
snake	0.20548		
black	0.20246		

Table 21: Similarity scores for the *nails* and *mushrooms* (fungi) classes from ImageNet-A.

Landbirds		*Waterbirds*	
Keyword	Cosine Similarity	Keyword	Cosine Similarity
plumage	0.21741	seagull	0.25743
feathers	0.20849	coastal	0.24091
		sea	0.22931

Table 22: Ablation study on a completely balanced version of Waterbirds.

k = 1		k = 5		k = 10		k = 25		k = 50	
Landbirds	*Waterbirds*	*Landbirds*	*Waterbirds*	*Landbirds*	*Waterbirds*	*Landbirds*	*Waterbirds*	*Landbirds*	*Waterbirds*
tree: 0.27 ± 0.01	cranes: 0.34 ± 0.01	foliage: 0.37 ± 0.02	ocean: 0.46 ± 0.01	tree: 0.37 ± 0.01	sea: 0.54 ± 0.01	tree: 0.36 ± 0.01	sea: 0.58 ± 0.02	foliage: 0.37 ± 0.00	sea: 0.56 ± 0.01
branches: 0.21 ± 0.00	lake: 0.31 ± 0.02	shrubs: 0.35 ± 0.02	beach: 0.37 ± 0.01	forest: 0.35 ± 0.02	ocean: 0.51 ± 0.00	forest: 0.36 ± 0.01	ocean: 0.56 ± 0.02	forest: 0.37 ± 0.01	ocean: 0.55 ± 0.01
brown: 0.19 ± 0.13	lagoon: 0.31 ± 0.01	forest: 0.34 ± 0.03	water: 0.34 ± 0.01	trees: 0.33 ± 0.03	beach: 0.44 ± 0.02	foliage: 0.36 ± 0.01	beach: 0.49 ± 0.02	trees: 0.33 ± 0.01	beach: 0.50 ± 0.01
trees: 0.18 ± 0.13	water: 0.27 ± 0.01	stalks: 0.33 ± 0.02	coastal: 0.32 ± 0.02	branch: 0.33 ± 0.00	coastal: 0.38 ± 0.01	trees: 0.33 ± 0.01	coastal: 0.43 ± 0.02	branch: 0.31 ± 0.00	shore: 0.44 ± 0.00
colors: 0.17 ± 0.12	sea: 0.26 ± 0.18	tree: 0.33 ± 0.03	lake: 0.30 ± 0.02	vegetation: 0.27 ± 0.02	water: 0.34 ± 0.02	branch: 0.31 ± 0.01	water: 0.37 ± 0.02	vegetation: 0.29 ± 0.00	water: 0.36 ± 0.01
leaves: 0.16 ± 0.12	pier: 0.25 ± 0.18	trees: 0.32 ± 0.03	shore: 0.26 ± 0.18	leaves: 0.24 ± 0.03	waves: 0.28 ± 0.00	vegetation: 0.27 ± 0.01	waves: 0.32 ± 0.01	leaves: 0.25 ± 0.01	waves: 0.33 ± 0.01
black: 0.13 ± 0.10	shoreline: 0.25 ± 0.17	vegetation: 0.28 ± 0.02	waves: 0.24 ± 0.00	shrubs: 0.23 ± 0.16	shoreline: 0.28 ± 0.20	plants: 0.21 ± 0.15	lake: 0.29 ± 0.02	shrubs: 0.24 ± 0.17	lake: 0.27 ± 0.01
foliage: 0.09 ± 0.13	crane: 0.21 ± 0.00	leaves: 0.28 ± 0.02	pond: 0.23 ± 0.16	foliage: 0.22 ± 0.15	midflight: 0.28 ± 0.00	forested: 0.11 ± 0.15	seagull: 0.27 ± 0.01	plants: 0.22 ± 0.15	midflight: 0.27 ± 0.01
forests: 0.09 ± 0.12	ship: 0.15 ± 0.21	branch: 0.27 ± 0.01	lagoon: 0.19 ± 0.13	stalks: 0.21 ± 0.15	shore: 0.27 ± 0.19	branches: 0.10 ± 0.14	midflight: 0.26 ± 0.01	forested: 0.11 ± 0.15	seagull: 0.15 ± 0.11
	seagull: 0.15 ± 0.10	plants: 0.22 ± 0.15	sea: 0.16 ± 0.22	forested: 0.12 ± 0.17	lake: 0.27 ± 0.03	branches: 0.10 ± 0.14	shore: 0.16 ± 0.22	branches: 0.10 ± 0.14	shoreline: 0.15 ± 0.21
	ocean: 0.13 ± 0.18	greenishbrown: 0.16 ± 0.12	seagull: 0.15 ± 0.11	greenery: 0.08 ± 0.11	boat: 0.22 ± 0.15				pond: 0.11 ± 0.15
	vessel: 0.12 ± 0.18	greenery: 0.16 ± 0.11	shoreline: 0.14 ± 0.19	greenishbrown: 0.08 ± 0.11	pond: 0.21 ± 0.15				

Table 23: Ablation study on Waterbirds, where we analyze the impact of the number of selected examples on the found keywords (k) and their respective similarity values.