ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Interpretable Approach to Discover and Remove Hidden Biases

Master's Thesis

Moritz Vandenhirtz

September 5, 2022

Department of Mathematics,
Seminar for Statistics, ETH Zürich

Supervisor: Prof. Dr. Julia E. Vogt

Advisors: Laura Manduchi, Ričards Marcinkevičs

Department of Computer Science, Institute for Machine Learning, ETH Zürich

Abstract

Spurious correlations are everywhere. While humans often do not perceive them, neural networks are notorious for learning unwanted correlations instead of the underlying decision rule. This difference brings rise to the problem of hidden bias, where a practitioner is unaware of the biased decision-making of its classifier. The deployment of such a biased model in the real world, where the spurious correlations might not hold anymore, would lead to unintended, adverse consequences. This thesis proposes a debiasing method that advances the research for alleviating this issue. We frame the problem in a probabilistic setting and propose a method that does not fall for unwanted spurious correlations. We use a variational autoencoder as our model's backbone and a theoretically founded reweighting scheme for training a biased and unbiased classifier. Using the unbiased classifier, we match or improve upon the state-of-the-art debiasing methods. Furthermore, the presented algorithm does not rely on the presence of bias labels for each individual in the training or validation set. Additionally, we propose a latent adversarial perturbation for visualizing the bias that helps practitioners become aware of the spurious attributes.

Acknowledgement

First and foremost, I would like to thank my supervisor Professor Dr. Julia E. Vogt, as well as my advisors Laura Manduchi and Ričards Marcinkevičs for their oversight of this thesis. The weekly meetings with Laura and Ričards were vital for guiding the work to a thesis I am proud of. I am very grateful for their support, insights, guidance, and advice throughout the last six months. Our collaboration, as well as the general group environment and atmosphere shaped by Julia, are major reasons for why I am looking forward to joining the Medical Data Science Group.

My gratitude also goes to all people that supported me during this time. I am thankful to my wonderful partner, Franziska Heusser, parents, friends, flatmates, office buddies, and many more. They helped me keep a positive attitude during the whole process and supported me by discussing ideas and proofreading the final writing.

Last but not least, I would like to acknowledge all of the staff at the Seminar for Statistics and the Institute for Machine Learning. Throughout my two years of studies, I could profit from various modules' wide range of topics. Thanks to the freedom I had in choosing those subjects, I was able to develop and follow my passions.

Contents

Chapter 1

Introduction

There are no mistakes in life, just lessons. Unfortunately, in machine learning, current models often fall prey to learning unwanted spurious correlations while searching for dependencies between variables. In this thesis, we embrace such mistakes by creating an algorithm that learns from its failures to overcome this problem.

The learning of spurious correlations occurs because decision rules based on them lead to satisfactory performance on the training data, despite the fact that they do not embody the true underlying connections. This biased way of making decisions is not robust and does not generalize well outside the training distribution. Thus, there has been an emergence of research that aims to mitigate this problem. Researchers make various assumptions that help them identify the bias variable, which should not be used for the decision rule. While many methods assume and leverage the presence of a sample-specific bias variable, in this thesis, we do not make such a strong assumption.

There are two main explanations why an individual bias variable might not be present. First, it can be infeasible to label each data point due to financial, time, or other constraints. Second, the undesirable variable might be unknown. Therefore, it is not possible to label it. A classifier might utilize this bias variable for making its prediction instead of using the desired signal, that is, the reason a particular label is present in each image. This mistake can happen if the bias is spuriously correlated with the label and is easier to learn than the signal. This leads to the problem that a classifier that bases its decision on the bias for predicting the label will perform reasonably well on the data at hand but will not generalize well outside the training data where the spurious correlation is not necessarily present anymore. To better understand how such a situation might arise, let us study a setting that will be the running example throughout this thesis.

Consider a data set consisting of histopathological whole-slide images (WSI) with the goal of classifying whether a tumor is present in these images. Further, most images where a tumor is present originate from hospital A and most images without a tumor originate from hospital B. Here, the presence or absence of a tumor is the signal, while the hospital from which the image originates is the bias. This bias is easily detectable as hospitals do not have the same machines. Hence, the images will look slightly different. For such a data set, a classifier might learn how the images of each hospital differ instead of learning how a tumor looks. Because predicting positive for hospital A and negative for hospital B will result in a reasonably good performance on the training data. Unfortunately, such a decision rule would have issues classifying new samples for which this spurious correlation does not hold. The practical application of such an algorithm in other hospitals would have detrimental effects. In Subsection 6.4, we will analyze how well our method helps mitigate this problem.

Previous works have focused on applicability in data sets with known but unlabelled bias (Bahng et al., 2020; Clark et al., 2019; Nam et al., 2020; Liu et al., 2021). To evaluate the performance, they use their knowledge about the bias to create an unbiased test set and evaluate how much their method improves the accuracy on it. A downside of this performance-optimizing research is that they are often unable to or do not focus on visualizing the bias. Recall that our setting arises either by insufficient resources to label the bias for each image or by not knowing the bias. We believe that not being able to label a known bias occurs substantially less often than not knowing the bias because such ignorance can happen with each data set.

For evaluating their debiasing capabilities, previous methods rely solely on an unbiased test set, which means they can not operate in the latter setting of hidden bias as constructing such a data set requires knowledge of the bias. In our previous example, if the biasing difference of hospitals is unknown, it is difficult to evaluate whether an allegedly unbiased classifier is indeed not using a bias. Here, it is important that a debiasing algorithm interpretably visualizes with respect to which characteristic it became indifferent such that a practitioner can decide whether this is indeed a bias. Otherwise, for an adjusted data set where all hospital images look similar, i.e., there is no bias, a standard debiasing technique might lead to a classifier that is invariant to the signal as it mistakes it for the bias. Here, the algorithm would not be able to recognize tumors after the debiasing procedure as this is the only thing it can unlearn due to the absence of bias. Without interpretability, a practitioner would not realize that by the application of a debiasing algorithm, such a disaster happened. Additionally, previous works have tuned their architecture and hyperparameters on each data set individually, which is only possible in the presence of a bias-labeled validation set. We do not want to make such an assumption in order to create a more generally applicable method.

For the reasons mentioned above, this thesis aims to develop an algorithm that not only trains an unbiased classifier but also interpretably visualizes the uncovered bias and does not rely on architectural and hyperparameter tuning on a bias-labeled validation set. We believe this will help uncover previously unknown biases in existing data sets, which is not possible with previous methods. The exaggerated aspiration of this thesis is that after having constructed any data set, practitioners would apply our method to check whether there are any biases they do not know of. After having visualized the potential biases, they would better understand whether they need to make adaptions without which a standard classifier would be unwillingly biased.

1.1 Contributions

The key contribution of this thesis is developing a novel, interpretable debiasing method[1]. Within this work, the main contributions are as follows:

- Theoretical problem formalization including graphical model as well as explicit formulating of assumptions
- Theoretically grounded reweighting scheme for training an unbiased classifier
- Proposal of a method that does not require a bias-labeled validation and test set
- Interpretably visualizing bias via innovative latent adversarial perturbations

1.2 Organisation

The thesis is structured as follows:

1. **Related Work**: In Chapter 2, we introduce the reader to the solutions other works propose for training an unbiased classifier in the presence of spurious correlations. First, we provide an overview of methods requiring an explicit bias label for each data point. Second, we present works that, instead of individual bias labels, rely on leveraging knowledge about the manifestation of the bias. Finally, we offer a closer look at debiasing schemes solely relying on the finding that malignant bias is easier to learn than signal.

[1]The code is publicly available at https://github.com/mvandenhi/Interpretable-Debiasing.

2. **Background for Variational Autoencoders**: In Chapter 3, we give the reader background information regarding the Variational Autoencoder. This generative model will be the backbone of our method.
3. **Interpretably Removing Hidden Bias**: In Chapter 4, we present the main body of work from this thesis. We commence by formalizing the problem and discussing the assumptions that we make. Followingly, we show our novel approach to create a theoretically founded interpretable debiasing method.
4. **Experiments**: In Chapter 5, we acquaint the reader with the three data sets we are using to evaluate our performance and detail how we implement three different baselines. Additionally, we display the different measures we take into account to assess performance.
5. **Results**: In Chapter 6, we present and visualize the results obtained from the experiments for our method and the baselines.
6. **Discussion**: In Chapter 7, we discuss the innovations presented in this thesis and support them with the results of our experiments. We argue that previous works have neglected important aspects of the hidden bias setting and show how our method fills these gaps.
7. **Conclusion**: In Chapter 8, we summarize the advances and findings presented in this thesis and give an outlook on possible future work.

Chapter 2

Related Work

In many data sets, a biasing variable is present, which we do not want to use for the decision rule. For example, this problem can arise when we do not want to discriminate against protected groups or if there is a spuriously correlated covariate whose effect would not generalize outside the data set distribution. In this chapter, we will outline how previous works have tackled this problem of developing a decision rule that does not take the bias into account.

The setting can be divided into two subcategories that require different approaches. In Section 2.1, the presence of bias in each individual input is captured by a discrete variable, which can be used during training to guide the algorithm toward a bias-invariant decision rule. Conversely, in Section 2.2, such an indicator is not given, and the method itself needs to recognize the bias and find a decision rule that does not rely on it.

2.1 Debiasing with Explicit Knowledge about Bias

In this section, we assume that each data point is defined by a triple $(\mathbf{x}, b, y)$, where $b(\mathbf{x}) \in \mathcal{B}$ is an *observed* bias attribute that is spuriously correlated with the input $\mathbf{x}$ and the label $y \in \mathcal{Y}$. A standard empirical risk minimizer predicting y from $\mathbf{x}$ will likely exploit the spurious correlation between the $m = |\mathcal{B}| \times |\mathcal{Y}|$ groups for its prediction. The resulting predictor is unfair towards groups for which this correlation does not hold. Additionally, even if the correlation is not spurious, in some circumstances, it is discriminating to make decisions based on the group affiliation of an individual. A famous example is the COMPAS Recidivism Algorithm, which according to Angwin et al. (2016), unfairly predicts a higher risk of recidivism for black defendants due to their race. Here, y is the risk of recidivism, while b describes the attribute race. While this specific finding is disputed (Flores et al., 2016;

Dressel and Farid, 2018), it is essential to develop measures to stop algorithms from learning such unfair decision rules.

To tackle this problem, standard group distributionally robust optimization (group DRO) aims to learn a model that minimizes the worst-case training loss over the m groups (Namkoong and Duchi, 2016). Specifically, they try to minimize the supremum of the training loss over all m groups, hopefully leading to a comparable performance in all groups. Even though this approach generalizes well for the classic (underparameterized) setting, generalization does not occur for certain groups in the overparameterized regime. This difference arises because for underrepresented groups, the abundance of parameters makes it easier to overfit on the scarce training data of these groups than learning the underlying structure. Sagawa et al. (2020) improve upon the existing literature on group DRO by combining overparameterized group DRO models with strong regularizers such as early stopping and an ℓ_2 penalty. This regularization prevents the algorithm from overfitting and leads to better worst-group generalization.

A different approach is to learn latent representations that contain as little information as possible with respect to the bias attribute. For example, Li and Vasconcelos (2019) try to create unbiased representations by adversarial sample reweighting. They dynamically assign each training data point a resampling weight through an optimization problem that tries to minimize the degree of bias contained in the latent representations. With the same goal in mind, several approaches try to achieve this by training an auxiliary bias-predicting classifier (Edwards and Storkey, 2016; Beutel et al., 2017; Kim et al., 2019). They use this classifier in an adversarial fashion to construct representations from which said classifier is unable to predict the bias. Simultaneously, they train a standard label-predicting classifier with the same latent variables, causing the representations to be informative with respect to the label but uninformative with respect to the bias. Similarly, Zhang et al. (2018) proof that the auxiliary classifier can be designed to enforce common fairness metrics such as demographic parity, equality of odds, or equality of opportunity as defined in Hardt et al. (2016).

Kearns et al. (2018) argue that any statistical notion of high-level fairness, such as in Hardt et al. (2016), can have some downfalls when taking subgroup conjunctions into account. For example, in a balanced data set, by predicting positive for all rich whites and poor blacks, we achieve demographic parity for the groups "race" and "wealth" separately, while this decision rule is maximally unfair when considering any conjunction of both groups. To solve this problem Creager et al. (2019) focus on the setting where the choice of sensitive bias attributes is not specified when training their model. After training, depending on the desired protected groups, they can easily modify their flexibly fair representation to achieve demographic parity for the different subgroups. The idea is to learn disentangled latent variables, some of which correspond to the sensitive attributes while others contain non-sensitive attributes. Then, depending on which sensitive demographic groups are chosen, a simple multilayer perceptron (MLP) can be trained on the subset of the representations not corresponding to the protected groups.

2.2 Debiasing without Explicit Knowledge about Bias

In many situations, it is unfeasible or impossible to obtain an explicit bias indicator for each sample. Hence, recent effort has been put into developing methods that do not rely on observing the sensitive attribute. In order to still mitigate the bias, different supportive assumptions about said variable are being made. In Subsection 2.2.1, while there is no observed sample-specific bias variable, the type of bias is implicitly known, such that a model can be designed to specifically target its manifestation. On the other hand, in Subsection 2.2.2, the assumptions made are broader as we do not require implicit knowledge about the bias type and often focus on the easiness of learning the bias compared to the signal (Nam et al., 2020).

2.2.1 Debiasing with Implicit Knowledge about Bias

Weinzaepfel and Rogez (2021) show that video action recognition models are biased toward leveraging the scene rather than the action. For instance, surfing is recognized by the waves of the sea rather than the surfer and surfboard. Similarly, Geirhos et al. (2019) notice that ImageNet-trained classifiers can exhibit a bias towards texture. While they use data augmentation to mitigate the bias, methods in this subsection make use of the bias inclination of such a classifier to train additional models that do not take the bias into account. While these methods do not require knowledge regarding the bias manifestation for each data point, they assume an understanding of the bias in order to design bias-capturing architectures. We term this understanding as implicit knowledge about the bias.

Bahng et al. (2020) do not rely on explicit knowledge about the bias and instead define a set of models designed to be biased towards the implicitly known bias attribute. For example, in the presence of texture bias, they use convolutional neural networks (CNN) with a small receptive field. Hence, they are more prone to focus on small-scale patterns such as texture. Simultaneously, they train a second classifier, which they optimize in a manner that learns to be invariant with respect to the biased classifier's decision rule. In the end, they keep the second, bias-invariant classifier to make unbiased predictions.

Clark et al. (2019) split the training into two parts. First, they train the biased classifier. Second, they train an unbiased classifier by combining its prediction with the prediction of the frozen biased classifier. As this combined model is already using the bias for making predictions, they expect the unbiased classifier to pick up on alternative, unbiased decision rules during training. For evaluation, they then only use the unbiased classifier.

Lastly, Wang et al. (2019) focus on resolving texture bias. They introduce a differentiable neural network building block specifically designed to extract textural information from images. They first train this network to capture the texture. Then, they train a second model by projecting its latent variables onto a subspace that is orthogonal to the representations of the first network. Hence, encouraging a decision rule that does not make use of the implicit textural bias attribute.

2.2.2 Debiasing without Implicit Knowledge about Bias

While papers from Section 2.1 and Subsection 2.2.1 focus on leveraging some kind of knowledge about the bias present in the data set, here we utilize more general properties of bias. As the method introduced in our thesis operates in this setting too, we will present a closer look at what comparable works have done, starting with Nam et al. (2020) who introduced it.

Nam et al. (2020) argue that bias can be split into benign and malignant. A benign bias does not influence a classifier's performance because it is more complicated to learn than the true, underlying decision rule. On the other hand, malignant bias is learned first as it is easier to learn than the target attribute. Hence, the authors develop a training scheme coined "Debiasing by Learning from Failure (LfF)" for disregarding malignant bias when training a classifier. The idea is similar to those in the previous subsection: Train a biased classifier to be as biased as possible and simultaneously train an unbiased classifier on samples for which the biased classifier performs poorly. We will call samples *bias-aligned* if utilizing the bias as decision rule leads to the correct label. Otherwise, we will call them *bias-conflicting*. Hence, the goal is to train the biased classifier and use it to differentiate between bias-aligned and bias-conflicting samples such that we can upweigh bias-conflicting samples for the unbiased classifier. To emphasize bias-aligned samples for the biased classifier, they deviate from the standard cross entropy loss (CE) and instead utilize the generalized cross entropy loss (GCE) by Zhang and Sabuncu (2018):

$$GCE(\hat{y}_b, y) = \frac{1 - \hat{y}_b^q}{q}, \tag{2.1}$$

where $\hat{y}_b$ is the predicted probability of the correct label y according to the biased classifier and $q \in (0, 1]$ is a hyperparameter to control the strength of emphasis. The GCE is best understood by inspecting its derivative

$$\frac{\partial GCE(\hat{y}_b, y)}{\partial \boldsymbol{\theta}_b} = \hat{y}_b^q \frac{\partial CE(\hat{y}_b, y)}{\partial \boldsymbol{\theta}_b}, \tag{2.2}$$

where $\boldsymbol{\theta}_b$ are the optimizable neural network parameters. This loss makes use of their finding that malignant bias is learned first, as it puts more weight on samples that the biased classifier already predicts well, which will be the bias-aligned samples. Conversely, for training the unbiased classifier, they weigh each sample by the relative difficulty score (RDS)

$$RDS(\hat{y}_s, \hat{y}_b, y) = \frac{CE(\hat{y}_b, y)}{CE(\hat{y}_s, y) + CE(\hat{y}_b, y)}, \tag{2.3}$$

where $\hat{y}_s$ is the predicted probability of the correct label y according to the unbiased signal classifier. Intuitively, the RDS downweighs samples that the biased classifier predicts well. Hence, making the unbiased classifier focus on bias-conflicting samples.

In practice, before calculating the RDS, Nam et al. (2020) apply an exponential moving average on the CEs followed by a class-wise normalization where for each class, they divide by the class-wise maximum CE of the whole data set. Our experiments suggest that this maximum-normalization plays a significant role during training. Thus, developing a weighting scheme that more directly improves performance would be desirable. Although not explicitly mentioned, during their hyperparameter tuning, they assumed the presence of a validation set for which the bias is explicitly known for each sample. This implicit assumption can be deduced from the training details, where the fixed number of training epochs varies between different data sets. Additionally, Nam et al. (2020) "acknowledge that assessing the reduction of potential risks by the proposed scheme can be a challenge without specifically identifying the biases," which motivates the interpretable approach presented in this thesis.

Liu et al. (2021) slightly adapt the idea of Nam et al. (2020). Instead of training both classifiers simultaneously, they divide the training into two stages. First, the biased classifier is trained. Second, they train the unbiased classifier and upweigh all samples that were misclassified by the biased classifier. Hence, they also try to focus on training on the bias-conflicting samples. Notably, they are the only authors in this setting that transparently talk about the problem of hyperparameter tuning, which we will discuss as well in Section 7.1. For their experiments, they tune their hyperparameters by assuming the presence of a bias-labeled validation set.

DisEnt by Lee et al. (2021) builds upon Nam et al. (2020) by utilizing their training scheme to create a disentangled representation useful for feature augmentation. As motivation, they show that the diversity of training samples is an important factor in training. Instead of training the unbiased classifier only on the sparse bias-conflicting samples, they try to synthesize additional samples for which using the bias as decision rule does not work. Their algorithm works as follows: First Lee et al. (2021) train the base structure from Nam et al. (2020) using GCE and RDS to create disentangled representations where signal and bias dimensions can be separated. After a predetermined number of updates, they start to swap the bias dimensions of different samples to synthesize representations with the same signal but different bias. Hence, making the bias unusable as decision rule for the label because it originates from a different sample. They then train their classifiers on those representations as well as on the original samples.

A caveat of Lee et al. (2021) is that for the swapping to have an influence, the classifiers need to take signal as well as bias dimensions into account for their prediction. While for truly disentangled representations, the biased and unbiased classifiers should consider bias and signal dimensions separately. Additionally, this method increases the number of hyperparameters as it has more losses induced by the swapping. In the absence of a bias-labeled validation set, this would complicate matters. The authors implicitly make the assumption that such a validation set is present, as their hyperparameters differ for each data set configuration.

To improve upon the non-interpretability of Nam et al. (2020), Lee et al. (2021) propose to train a decoder ex-post to reconstruct the images given their latent representations. By repeating their swapping process and reconstructing the images after the swap, they expect to visualize the bias present in the data set as they can compare reconstructions with the same signal but swapped bias dimensions. While this is a step in the right direction, by training the decoder ex-post, they can not update the representations, which might not contain all relevant information necessary to reconstruct the image because the representations were optimized for classification. Also, by swapping the bias dimensions and then reconstructing, all residual information about the image that is present in the bias dimension gets swapped too. This thesis will present a more sophisticated approach for interpretably identifying the bias.

Kim et al. (2021) tackle the problem of interpretability as well. Similar to Liu et al. (2021), they train the biased classifier first. Then, they try to split the training set into bias-aligned and bias-conflicting images according to how well the biased classifier can predict the label. With this estimated split, they make use of the SwapAE (Park et al., 2020). This autoencoder takes two images as input. As output, it generates an image that maintains the first image's content and applies the second image's style. Thus, Kim et al. (2021) input a bias-aligned and a bias-conflicting image to generate a new image containing the bias-irrelevant features from the first and the bias-relevant features from the second image. To ensure that the SwapAE captures the bias-relevant features of the bias-conflicting image, they make it focus on the patches of the images that the biased classifier uses for its prediction as determined by its class activation map (Zhou et al., 2016). With this feature augmentation technique, they pursue the same goal as Lee et al. (2021), that is, to create more diverse bias-conflicting images. Finally, these new images are used to train their unbiased classifier. By using the class activation map, the technique of Kim et al. (2021) operates in the pixel space. While this is adequate for the data sets they used, it is unclear whether this approach generalizes to other biases. Additionally, by using the SwapAE, they assume that the content of an image is the signal while the bias is the image style. This assumption might not be fulfilled in all applications.

Lastly, Darlow et al. (2020) also aim to generate new diverse bias-conflicting images. They leverage a vector quantized variational autoencoder (van den Oord et al., 2017) and add a simple biased classifier on top of the latent representations. After training, they perturb the latent dimensions such that the biased classifier is as unsure as possible about the label. This procedure leads to a representation for which a classifier that looks at only the bias can not make a confident prediction. Hence, there is no bias present that could be used for a prediction. This perturbed representation is then passed through the decoder to generate images without relevant bias. Consequently, a second, unbiased classifier is trained on these images, which should not contain bias that can be used for predicting the label. A word of warning to the interested reader: While the results of Darlow et al. (2020) seem groundbreaking, they are achieved by utilizing a much more advanced neural network structure, such that the results are not directly comparable to the other works.

To summarize, with LfF, Nam et al. (2020) introduce the field of debiasing without knowledge about the bias and leverage their finding that malignant bias is easier to learn than the underlying signal attribute. Through the GCE, they train a biased classifier. Simultaneously they train an unbiased classifier by upweighting bias-conflicting samples through the RDS such that the classifier focuses on samples for which the bias leads to no valid decision rule. DisEnt by Lee et al. (2021) builds on top of LfF by swapping the latent bias vectors of different images while keeping the signal. These augmented, bias-conflicting representations are then passed to the unbiased classifier. With this procedure, the unbiased classifier becomes invariant to the bias. Following the augmentation idea of DisEnt, Kim et al. (2021) try to augment the training data by leveraging the SwapAE (Park et al., 2020) where they create images with similar signal but different biases. To identify the bias in an image, they leverage the class activation map (Zhou et al., 2016) of the biased classifier. Finally, Darlow et al. (2020) try to augment the training data by training a vector quantized variational autoencoder (van den Oord et al., 2017), adversarially perturbing the latent dimensions according to the biased classifier and reconstructing these debiased images.

Chapter 3

Background on Variational Autoencoders

This segment intends to familiarize the reader with the Variational Autoencoder (VAE) developed by Kingma and Welling (2014), which is the backbone of our method. We want to explicitly state that this is not our work and that we are merely summarizing the work of Kingma and Welling (2014) in this chapter. Nevertheless, in Subsection 4.2.1, among other things, we will present how we adapt their work to be applicable in our context.

The goal of a VAE is to learn the underlying data distribution of a given data set. By data distribution we mean a combination of latent variables we will denote by $\mathbf{z}$, which manifest themselves in observable attributes $\mathbf{x}$ described by $p(\mathbf{x}|\mathbf{z})$. For example, $\mathbf{z}$ could have the values *plane* and *sky*, for which $\mathbf{x}$ would describe the union of all pixels values from an image in which a human would recognize a flying aircraft. When training a machine learning model, it is desirable that it perceives its input $\mathbf{x}$ not as a concatenation of pixel values but instead understands the underlying concepts of the image. This is exactly what the VAE achieves by learning the data-generating latent representations. Access to this data distribution enables one to utilize it for a multitude of applications, such as generating new samples, detecting outliers, or label prediction.

A VAE is a probabilistic generative model that allows one to take the inherent variability of a data set into account. They are trained by maximizing a lower bound for the likelihood of the data, denoted by $p(\mathbf{x})$, which is sometimes called evidence. Specifically, they are maximizing the Evidence Lower Bound (ELBO):

$$\log[p(\mathbf{x})] \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\left[\log[p(\mathbf{x}|\mathbf{z})]\right] - D_{KL}\left[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right], \tag{3.1}$$

where p denotes the data's true distribution while q represents an approximation thereof and D_{KL} refers to the Kullback–Leibler divergence (Kullback and Leibler, 1951). For a detailed derivation, we refer to Appendix A.

In order to maximize the ELBO, one usually parametrizes $p(\mathbf{x}|\mathbf{z})$, as well as $q(\mathbf{z}|\mathbf{x})$ by a neural network (NN) that outputs the parameters of the defined distributions. Some simplifying assumptions are made on the covariance matrix to reduce the complexity. Conversely, $p(\mathbf{z})$ is parametrized by a distribution with fixed parameters as it is independent of the data at hand. A simplified illustration of a VAE can be found in Figure 3.1.

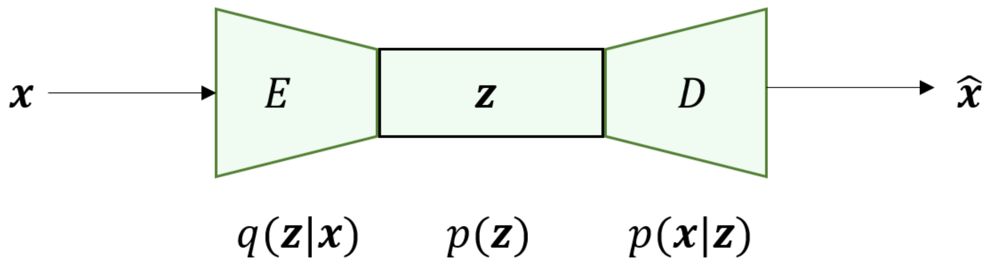


Figure 3.1: Graphical depiction of the structure of a Variational Autoencoder. The input $\mathbf{x}$ is passed through an encoder E that predicts the parameters of the approximate posterior $q(\mathbf{z}|\mathbf{x})$. We then sample from this distribution to obtain $\mathbf{z}$, which is constrained to be in the vicinity of the prior $p(\mathbf{z})$ by the ELBO. Lastly, $\mathbf{z}$ is passed through the decoder D that predicts $\hat{\mathbf{x}}$, which is the mean of $p(\mathbf{x}|\mathbf{z})$ to reconstruct $\mathbf{x}$ as well as possible.

Training a VAE is a two-step process, as is the case with most NN. In the first step, the input $\mathbf{x}$ is passed through an encoder E to predict the parameters of the approximate posterior $q(\mathbf{z}|\mathbf{x})$. Usually, $q(\mathbf{z}|\mathbf{x})$ is assumed to be a multivariate normal distribution with diagonal covariance matrix. Here, the encoder predicts the mean and variance of $q(\mathbf{z}|\mathbf{x})$. Then, a latent vector $\mathbf{z}$ is sampled from this multivariate normal distribution. $\mathbf{z}$ is then passed through the decoder D to recover the original input $\mathbf{x}$ as well as possible. Normally, this is done by predicting the means of $p(\mathbf{x}|\mathbf{z})$, usually assumed to be a multivariate normal distribution with a fixed diagonal covariance matrix.

Having access to all distributions required in the ELBO, in the second step, the weights of the neural networks are updated to increase the ELBO of the datum. This updating is done by gradient ascent or variants of it, which essentially looks at the derivative of the ELBO with respect to each weight and adjusts the weight in the direction of increasing the ELBO. Such a procedure requires that it is possible to compute gradients of the randomly sampled $\mathbf{z}$ with respect to the parameters of the normal distribution $q(\mathbf{z}|\mathbf{x})$.

Naively, it is not possible to calculate the derivative of a realization with respect to its distributional parameters. Kingma and Welling (2014) propose the reparameterization trick that makes use of the fact that for $z \sim \mathcal{N}(\mu, \sigma^2)$

we have that $z = \mu + \sigma\varepsilon$ for $\varepsilon \sim \mathcal{N}(0, 1)$. Thus, they sample from a standard normal distribution and linearly transform this sample by the predicted parameters to obtain **z**. This reparameterization makes it possible to take the derivative of **z** with respect to the parameters, as they are connected by a linear transformation.

Intuitively, the second term of the ELBO formulation in Expression 3.1 works as a regularizer and tries to bunch all latent points together. Adversely, the first term, often referred to as reconstruction loss, has the goal of reconstructing the original input as well as possible, which requires the latent representation to be informative. The clash of both losses creates a regularized latent space that reduces the dimensionality of the data while preserving informativeness with respect to the data. This informative reduction is achieved by structuring the information and variability of **x** in the latent representation **z**.

Chapter 4

Interpretably Removing Hidden Bias

This chapter presents the main body of work from this thesis. First, in Section 4.1, we introduce the framework in which our model is expected to operate. Then, following the theoretical foundation, in Section 4.2, we set forth our interpretable method to train an unbiased classifier.

4.1 Problem Formalization

Before presenting our method in the next section, we will define the framework of the proposed approach by highlighting the underlying assumptions. In doing so, we hope to shed light on the situations in which our or related methods should be applied. Our theoretical framework improves upon the existing literature, which focuses more on the applied side of the problem. First, in Subsection 4.1.1, we define the graphical model, which we assume to be the underlying data generating mechanism for data sets on which our debiasing method is applicable. Afterward, in Subsection 4.1.2, we explicitly state the assumptions we require to enable the debiasing capabilities of our model.

4.1.1 Graphical Model

Consider a data set with observed features $\mathbf{x}$ and observed label y. Here, we discuss by what unobserved data generating mechanism $p(\mathbf{x}, y)$ we expect the data to have been generated such that an empirical risk minimizer might accidentally learn the bias. While works discussed in Section 2.1 assume the presence of a sample-specific bias indicator b, we do not require the bias attribute to be observed. Instead, we allow it to be an unobserved latent set of variables and denote it as $\mathbf{z}_b$. Additionally to the latent bias variables $\mathbf{z}_b$, we necessarily require the existence of a latent set of signal variables $\mathbf{z}_s$, which are the variables responsible for the observed label y and should be leveraged for learning the underlying decision rule. The input $\mathbf{x}$ is generated by a combination of both latent variables $\mathbf{z}_b$ and $\mathbf{z}_s$. The connection from $\mathbf{z}_b$ to $\mathbf{x}$ allows the classifier to exploit the bias $\mathbf{z}_b$ for its decision rule. The connection from $\mathbf{z}_s$ to $\mathbf{x}$ allows our method to learn an unbiased classifier that focuses on $\mathbf{z}_s$ instead of $\mathbf{z}_b$ when looking at the input $\mathbf{x}$. With the desiderata at hand, a classifier would have no intention of learning $\mathbf{z}_b$ as it has no predictive value for y. Hence, we introduce the unobserved binary sample selection variable r. This variable can be thought of as a rejection sampler depending on $\mathbf{z}_s$ and $\mathbf{z}_b$ that decides whether a given data point will be observed at all. We present the full graphical model in Figure 4.1.

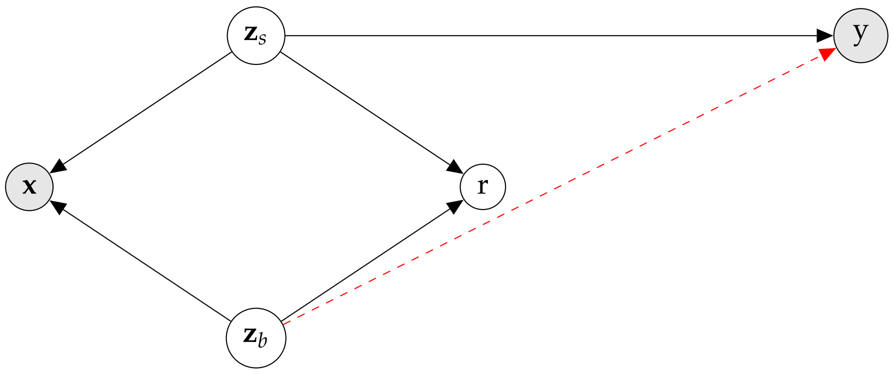


Figure 4.1: Graphical model for the setting in which our method is expected to operate. Grey nodes are observed variables, while white nodes are unobserved. Black arrows signify a causal relationship, while the red dashed arrow indicates a spurious correlation when conditioning on r.

The binary variable r is the key component of the graphical model, which is the reason why a training set is biased and gives rise to the biased classifier. If $r = 1$, then a given data point is observed and included in the data set. Accordingly, $r = 0$ means this constellation of variables was not observed and hence can not be included in the data set. This implies that by training a classifier with a training set following this graphical model, we are implicitly conditioning on the variable r, as we are only training on data points that have been observed. We call a sample *bias-aligned*, if it has a combination of $\mathbf{z}_s$ and $\mathbf{z}_b$ such that $p(r = 1|\mathbf{z}_s, \mathbf{z}_b)$ is high. Conversely, if $p(r = 1|\mathbf{z}_s, \mathbf{z}_b)$ is low, such a sample is called *bias-conflicting*. Therefore, while neither $r = 1$ deterministically implies bias-aligned nor vice versa, the two often coincide. This concurrence leads to a training set with many bias-aligned images that give rise to a classifier using the biased decision rule.

Through the conditioning on r, which is done implicitly by using the training set, an indirect dependence from $\mathbf{z}_b$ to $\mathbf{z}_s$ and consequently to y emerges. This spurious connection is the reason why $\mathbf{z}_b$ can be used as a biased decision rule that still gives high accuracy for classifying y. In a data set following this graphical model, bias-aligned samples are more prevalent than if there was no variable r in the graphical model. This means that certain combinations of $\mathbf{z}_s$ and $\mathbf{z}_b$ will appear more. Thus, a classifier can learn to recognize bias $\mathbf{z}_b$ and through this infer the signal $\mathbf{z}_s$ and label y, which will be a correct conclusion for the many bias-aligned samples in the data set. Nevertheless, as previously mentioned, applying such a classifier in an environment where the connection through r no longer holds can have detrimental effects. This is why we want to train an unbiased classifier that directly recognizes $\mathbf{z}_s$. Because understanding the variable r is vital, we study it in the context of our running example to clarify potential uncertainties.

Recall the running example from Chapter 1, in which hospital A produces mostly tumorous images while hospital B has many tumor-free images. A classifier might learn to differentiate hospitals, instead of the manifestation of a tumor, for predicting the presence of a tumor. Thus, $\mathbf{z}_b$ describes the different hospitals, while $\mathbf{z}_s$ represents the presence or absence of a tumor. Here, r describes the fact that bias-aligned samples with $\mathbf{z}_s = tumor$ and $\mathbf{z}_b = hospitalA$ are observed a number of times, as $p(r = 1|\mathbf{z}_s = tumor, \mathbf{z}_b = hospitalA)$ is high. This increased probability of observing both attributes together might arise due to hospital A being specialized in tumors. As such, patients with tumors from other hospitals are sent there for in-depth analysis. Still, predicting that each patient from this hospital has a tumor is not a sensible decision rule, which is what the method presented in this thesis tries to prevent.

4.1.2 Assumptions

Knowing what assumptions are implicitly made when applying a method is often overlooked. Even though this knowledge is vital for understanding the framework and interpretation of results. In an effort towards more transparency, we will introduce the assumptions that are made when our model is used and what the effects are if these are not fulfilled.

1. **Graphical model holds**: The most important assumption made is that the graphical model specified in Subsection 4.1.1 describes the underlying data generating mechanism. Similar to one of the assumptions of linear regression, which states that the model is correctly specified, this is a hypothesis that, for any practical data set, is not testable. One has to accept the aphorism by Box (1979) that states ”all models are wrong, but some are useful.” By this, we mean that if the graphical model is a reasonably good description of the underlying data-generating process, then it can be expected that our method will perform reasonably well.

2. **Bias is easier to learn than signal**: In order to establish an environment where a debiasing method is necessary, we introduce the assumption that for a standard classifier, learning the biased decision rule is easier than the underlying signal-based decision rule. As neural networks

prefer to learn simple patterns first (Arpit et al., 2017), this assumption makes it such that a standard classifier latches onto the easy-to-learn bias instead of the signal. Without this assumption, a standard classifier might learn the signal, which would preclude the need for a debiasing technique. We subdivide this assumption into three parts.

a) In order for the bias to be easier to learn than the signal, we assume that the relationship from $\mathbf{x}$ to y through $\mathbf{z}_b$ is easier to capture than through $\mathbf{z}_s$.

b) Additionally, we require for bias-aligned samples that the biased decision rule contains the same predictive power as if utilizing the signal. This means that for bias-aligned samples $p(y|\mathbf{z}_s) = p(y|\mathbf{z}_b)\ \forall \mathbf{x}$.

c) Lastly, we assume that there are enough bias-aligned samples such that the easiness of learning the relationship of $\mathbf{x}$ and y through $\mathbf{z}_b$ is not outweighed by the fact that this decision rule does not work for the bias-conflicting samples.

Combined, these three subassumptions indicate that a standard classifier will focus on learning the bias rather than the signal for its decision rule, as it prefers simple, biased patterns as long as it is not losing its predictive power for the bulk of samples.

3. **Presence of bias-conflicting samples**: While most of the data set consists of samples for which utilizing the bias will result in the correct label, we require the presence of data points for which this decision rule does not lead to the correct solution. The reason for this is twofold. First, without this assumption, both decision rules, based on either bias or signal, would be valid as they predict the label as well as possible. Hence, it is ambiguous which is the correct solution. Second, the method we present in Section 4.2 operates by upweighting bias-conflicting samples, which is not possible if none exist.

If any of these assumptions do not hold, then the biased classifier we introduce in Subsection 4.2.1 will not be able to learn the bias. Consequently, we would not be able to train an unbiased classifier, and our method would not work. As usual, it is not possible to check all of our assumptions a priori. We hope that by our interpretable visualizations presented in Subsection 4.2.2, a practitioner can assess a posteriori whether any assumptions are violated. For example, by visualizing what the model believes is the manifestation of the bias, it is possible to determine whether there exists any easy-to-learn bias at all in the data set.

4.2 Method

In the following subsections, we present the method developed in this thesis, which aims to alleviate the problem of hidden bias in a data set corresponding to the setting outlined in Section 4.1. In Chapter 3, we acquainted the reader with the Variational Autoencoder, which will be the backbone for creating interpretable visualizations of the bias. In order to visualize said bias, we need to disentangle it from the signal. Therefore, we present our two-model reweighting structure to single out the bias in Subsection 4.2.1. During the disentangling of signal from bias, we will simultaneously create an unbiased signal classifier that does not take the bias into account for its predictions. Then, in Subsection 4.2.2, we introduce the algorithm we use for interpretably visualizing the bias from our given model structure. Finally, in Subsection 4.2.3, we present the specifications we use for running our method, with the aim of having a structure that does not need any tuning on a specific data set.

4.2.1 Biased and Unbiased Classifier

The original VAE (Kingma and Welling, 2014) was designed for unsupervised learning. In this thesis, we extend the VAE to include the reweighting structure of LfF (Nam et al., 2020). This extension results in a supervised method that contains an unbiased classifier as well as an interpretable visualization mechanism. We choose a VAE over other generative models for its mathematically founded ELBO, allowing us to encode our graphical model in the loss. In turn, this decreases the number of hyperparameters because the ELBO determines the relative weight of the losses. Lee et al. (2021) suggested training the decoder D on the latent embedding only after having trained the unbiased classifier and having fixed the weights of the encoder E. We believe such training needs to happen simultaneously. Otherwise, if the encoding of the latent dimensions is only trained to optimize the classifier's performance, they will not contain information irrelevant to the label but relevant to the image. This would make the reconstruction of the original image a challenging task.

We will now step-by-step integrate the assumptions made in Subsection 4.1.2 into the ELBO for deriving a loss function for our VAE. First, we integrate Assumption 1 and adapt the VAE to the structure of our graphical model outlined in Subsection 4.1.1 for which we take inspiration from Kingma et al. (2014). The key difference to the original ELBO is the inclusion of the observed label y as well as having separate latent variables $\mathbf{z}_s$ and $\mathbf{z}_b$.

$$
\begin{aligned}
\log[p(\mathbf{x},y)] &= \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log[p(\mathbf{x},y)]d\mathbf{z}_s d\mathbf{z}_b \\
&= \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log\Big[\frac{p(\mathbf{x},y,\mathbf{z}_s,\mathbf{z}_b)}{p(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)}\frac{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)}{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)}\Big]d\mathbf{z}_s d\mathbf{z}_b \\
&= \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log\Big[\frac{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)}{p(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)}\Big]d\mathbf{z}_s d\mathbf{z}_b \\
&\quad + \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log\Big[\frac{p(\mathbf{x},y,\mathbf{z}_s,\mathbf{z}_b)}{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)}\Big]d\mathbf{z}_s d\mathbf{z}_b \\
\log[p(\mathbf{x},y)] &\geq \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log\Big[\frac{p(\mathbf{x},y,\mathbf{z}_s,\mathbf{z}_b)}{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)}\Big]d\mathbf{z}_s d\mathbf{z}_b \\
&= \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log\Big[\frac{p(\mathbf{x},y|\mathbf{z}_s,\mathbf{z}_b)p(\mathbf{z}_s,\mathbf{z}_b)}{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)}\Big]d\mathbf{z}_s d\mathbf{z}_b \qquad (4.1) \\
&= \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log[p(\mathbf{x},y|\mathbf{z}_s,\mathbf{z}_b)]d\mathbf{z}_s d\mathbf{z}_b \\
&\quad - \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log\Big[\frac{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)}{p(\mathbf{z}_s,\mathbf{z}_b)}\Big]d\mathbf{z}_s d\mathbf{z}_b \\
&= \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log[p(y|\mathbf{z}_s)p(\mathbf{x}|\mathbf{z}_s,\mathbf{z}_b)]d\mathbf{z}_s d\mathbf{z}_b \\
&\quad - \iint q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x},y)\log\Big[\frac{q(\mathbf{z}_s|\mathbf{x})q(\mathbf{z}_b|\mathbf{x})}{p(\mathbf{z}_s)p(\mathbf{z}_b)}\Big]d\mathbf{z}_s d\mathbf{z}_b \\
&= \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[\log[p(\mathbf{x}|\mathbf{z}_s,\mathbf{z}_b)]\right] - D_{KL}\left[q(\mathbf{z}_s|\mathbf{x})q(\mathbf{z}_b|\mathbf{x})||p(\mathbf{z}_s)p(\mathbf{z}_b)\right] \\
&\quad + \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[\log[p(y|\mathbf{z}_s)]\right]
\end{aligned}
$$

Starting from the original ELBO derivation in Appendix A, in Expression 4.1 we expand the observed variables to be $\mathbf{x}$ and y as well as separating $\mathbf{z}$ into $\mathbf{z}_s$ and $\mathbf{z}_b$. We then insert Assumption 1 about our graphical model in the penultimate line of Expression 4.1. Note that the first two terms of the final inequality are similar to the original ELBO. The difference is only the division of $\mathbf{z}$ into $\mathbf{z}_s$ and $\mathbf{z}_b$. While the last term is introducing a classifier that takes the latent signal attributes as input and predicts the label from them. We do not include learning the unobserved variable r and treat it as a nuisance parameter. Learning this parameter is as hard as learning the disentangled $\mathbf{z}_s$ and $\mathbf{z}_b$ because its value is determined through their separate realizations that need to be learned. Therefore trying to learn it explicitly grants no additional benefit to our current setup.

So far, this ELBO only justifies using a single classifier on the signal dimensions. We will now integrate the easy-to-learn Assumption 2 to account for the fact that a simple classifier would learn the bias instead of the signal in its latent dimensions. Nam et al. (2020) have shown that training with a bias and a signal classifier helps disentangle the latent representations, which is what we will incorporate in the ELBO.

Recall the easy-to-learn Subassumption 2b, which states that for bias-aligned samples it holds that $p(y|\mathbf{z}_s) = p(y|\mathbf{z}_b)\ \forall \mathbf{x}$ and consequently $\log[p(y|\mathbf{z}_s)] = \log[p(y|\mathbf{z}_b)]\ \forall \mathbf{x}$. Note that this equation only holds for the true data-generating latent set of variables and not necessarily for the estimated embeddings of our encoder. The samples for which a biased classifier can predict the label well are the ones that are bias-aligned. To measure this, we introduce the variable $\hat{y}_b$, the predicted probability of the correct label y by a biased classifier. $\hat{y}_b$ is proportional to the bias alignedness of a sample as it captures how well the bias can be leveraged for inferring the signal and the derived label. We use this to get

$$\hat{y}_b \log[p(y|\mathbf{z}_s)] \approx \hat{y}_b \log[p(y|\mathbf{z}_b)]\ \forall \mathbf{x}. \tag{4.2}$$

For bias-aligned samples, a biased classifier can predict y using z_b, and we have $\hat{y}_b \to 1$. This implies $\log[p(y|\mathbf{z}_s)] \approx \log[p(y|\mathbf{z}_b)]$, which holds as it fulfills the original easy-to-learn assumption. Conversely, for bias-conflicting samples, a biased classifier predicts the wrong class, and we have that $\hat{y}_b \to 0$. Thus, we arrive at $0 \approx 0$, which also holds.

We integrate this finding in the ELBO of Expression 4.1

$$\begin{aligned}
\log[p(\mathbf{x}, y)] &\geq \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[\log[p(\mathbf{x}|\mathbf{z}_s, \mathbf{z}_b)]\right] - D_{KL}\left[q(\mathbf{z}_s|\mathbf{x})q(\mathbf{z}_b|\mathbf{x})||p(\mathbf{z}_s)p(\mathbf{z}_b)\right] \\
&\quad + \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[\log[p(y|\mathbf{z}_s)]\right] \\
&= \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[\log[p(\mathbf{x}|\mathbf{z}_s, \mathbf{z}_b)]\right] - D_{KL}\left[q(\mathbf{z}_s|\mathbf{x})q(\mathbf{z}_b|\mathbf{x})||p(\mathbf{z}_s)p(\mathbf{z}_b)\right] \\
&\quad + \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[(1 - \hat{y}_b + \hat{y}_b)\log[p(y|\mathbf{z}_s)]\right] \\
&\approx \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[\log[p(\mathbf{x}|\mathbf{z}_s, \mathbf{z}_b)]\right] - D_{KL}\left[q(\mathbf{z}_s|\mathbf{x})q(\mathbf{z}_b|\mathbf{x})||p(\mathbf{z}_s)p(\mathbf{z}_b)\right] \\
&\quad + \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[(1 - \hat{y}_b)\log[p(y|\mathbf{z}_s)] + \hat{y}_b\log[p(y|\mathbf{z}_b)]\right],
\end{aligned} \tag{4.3}$$

where we have inserted Expression 4.2 in the last line. In the last term of the final inequality, we recover a particular case of the Generalized Cross Entropy Loss (Zhang and Sabuncu, 2018) used by Nam et al. (2020). According to them, the classifier $p(y|\mathbf{z}_b)$ trained with this loss will become biased as a result of upweighting biased samples. Therefore, we can leverage this biased classifier for calculating $\hat{y}_b$. As this biased classifier is trained simultaneously with the weighting of the samples, it will not be perfectly biased. Additionally, $\hat{y}_b$ is merely an approximation of bias alignedness. To account for this, we relativize its prediction by the hyperparameter $q \in (0, 1]$ and have fully recovered the GCE loss. The final ELBO we will be optimizing is as follows:

$$\begin{aligned}
\log[p(\mathbf{x}, y)] &\geq \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[\log[p(\mathbf{x}|\mathbf{z}_s, \mathbf{z}_b)]\right] - D_{KL}\left[q(\mathbf{z}_s|\mathbf{x})q(\mathbf{z}_b|\mathbf{x})||p(\mathbf{z}_s)p(\mathbf{z}_b)\right] \\
&\quad + \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[(1 - \hat{y}_b)^q\log[p(y|\mathbf{z}_s)] + \hat{y}_b^q\log[p(y|\mathbf{z}_b)]\right]
\end{aligned} \tag{4.4}$$

The reasoning mentioned beforehand serves the purpose of justifying and integrating a bias classifier into the model structure. Otherwise, following the graphical model, one would expect solely a signal classifier, which does not allow for disentangling the latent space. Thus, there is a need for mathematically integrating the easy-to-learn assumption, which gives rise to the need for debiasing. We want to transparently state a critical point of the derivations such that future work can improve upon our considerations: The assumption underlying approximate Equality 4.2 holds for all $\mathbf{x}$, which implies that it holds over the true posterior $p(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x})$. However, Expression 4.3 inserts the approximate Equality 4.2 inside of a double integral over approximate posterior $q(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x})$. The problem is that $q(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x})$ is merely an approximation of $p(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x})$ and therefore they might differ. Especially during training, $q(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x})$ might be far apart from the true posterior.

Having derived the ELBO for our graphical model, in Figure 4.2, we depict the model structure proposed in this thesis that integrates this optimizable expression. By introducing the easy-to-learn assumption into the ELBO, we have theoretically justified the use of a bias classifier C_b additionally to the unbiased signal classifier C_s. The purpose of having two classifiers is to disentangle the latent bias and signal representations. While Nam et al. (2020) train the signal classifier by utilizing their relative difficulty score, we derive a much simpler reweighting scheme using $(1 - \hat{y}_b)^q$. This weight resembles the focal loss introduced by Lin et al. (2017), with the difference that they do not use one model's prediction for weighting another model but work only with one model. Thus, they also backpropagate with respect to the reweighting while we detach this factor to not mix the gradients of both classifiers. Lin et al. (2017) show that the focal loss focuses training on hard examples. To be specific to our setting, it focuses on data points that are hard for the biased classifier. Training two classifiers simultaneously while weighting training samples differently is key to disentangling the signal and bias contained in the image.

The bias classifier C_b focuses on learning easy samples since the Generalized Cross Entropy Loss upweighs samples that are already being classified well. Due to Assumption 2, these are likely to be the bias-aligned samples causing C_b to learn the bias attributes $\mathbf{z}_b$. Opposed to this, the signal classifier focuses on samples that are not being predicted well by a bias classifier that focuses on easy-to-learn samples. These are likely to be bias-conflicting data points for which the biased decision rule does not work. Consequently, C_s will learn the signal $\mathbf{z}_s$ as for these samples utilizing $\mathbf{z}_b$ does not lead to the correct prediction. This reweighting scheme explains the necessity of Assumption 3 because there would not be any data from which the signal classifier can learn in the absence of bias-conflicting samples.

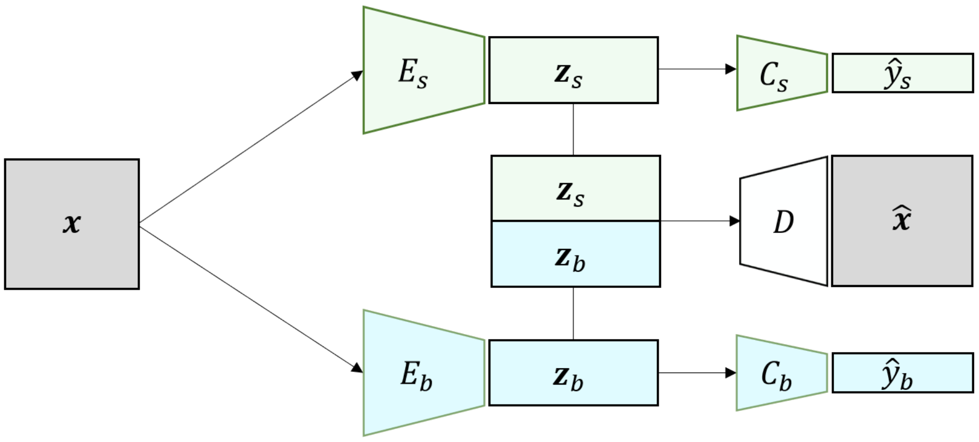


Figure 4.2: Graphical overview of our proposed model structure. The input $\mathbf{x}$ is passed through the signal and bias encoders E_s & E_b to obtain the latent signal and bias embeddings $\mathbf{z}_s$ & $\mathbf{z}_b$. These representations are then passed through their respective classifier C_s & C_b to predict the probability of the correct class of y by $\hat{y}_s$ & $\hat{y}_b$. Lastly, the latent attributes are concatenated and passed through the decoder D to reconstruct the original image $\mathbf{x}$ as well as possible by $\hat{\mathbf{x}}$.

The decoder supports the classifiers in condensing the information contained in the input and makes the learning of the latent attributes easier. Furthermore, it helps find potentially challenging parts of the signal as it requires the model to learn all aspects of an image. Additionally, it prevents E_s from focusing on learning bias too, because, to reconstruct the image, the signal attributes must be learned somewhere. Through this model structure, we obtain a disentangled latent space as well as an unbiased classifier that makes predictions according to the signal in the input.

4.2.2 Latent Adversarial Perturbation

After training our model, we leverage the decoder to interpretably visualize the bias. For this, we propose an innovative latent adversarial perturbation based on Deepfool by Moosavi-Dezfooli et al. (2016), which we will introduce shortly. Deepfool generates adversarial perturbations to fool a classifier into predicting the wrong label while modifying the input as little as possible. The space of possible perturbations is infinite. Thus they have to make approximations to reduce the computational complexity. Moosavi-Dezfooli et al. (2016) point out that for a binary, affine classifier $C(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + \mathbf{b}$ the minimal perturbation δ for which $\mathbf{x} + \delta$ fools the classifier is given by $\delta = -\frac{C(\mathbf{x})}{||\mathbf{w}||_2^2}\mathbf{w}$. For a general binary classifier they adopt an iterative scheme of linearizing the classifier around the current point $\mathbf{x}_i$ and then computing the minimal perturbation for the linearized classifier as

$$\delta_i = -\frac{C(\mathbf{x}_i)}{||\nabla C(\mathbf{x}_i)||_2^2}\nabla C(\mathbf{x}_i).$$

As this perturbation is an approximation, it might not change the predicted label of the classifier. Therefore, this process is repeated until the classifier is fooled and predicts the wrong label.

Starting from DeepFool (Moosavi-Dezfooli et al., 2016), which takes the input $\mathbf{x}$ and predicts y using the classifier $C(\mathbf{x})$, we adjust this algorithm to instead perturb $\mathbf{z}_b$ with respect to the biased classifier C_b. As the goal is to visualize interpretably what the model believes to be the bias, we use it solely at inference time, when the latent space has been disentangled, and not during training. While Deepfool can be adapted to work for the multiclass case, for bias-aligned images, we recover the computationally less intensive binary case by randomly sampling a target label that is different from the true label.

With this, we generate the adversarially perturbed bias representation $\mathbf{z}_{b,adv}$ that is altered such that the biased classifier cannot predict the correct label. This implies that we removed the bias from $\mathbf{z}_b$ as this is what C_b uses for its predictions. $\mathbf{z}_{b,adv}$ is then concatenated with $\mathbf{z}_s$ and passed through the decoder to reconstruct $\hat{\mathbf{x}}_{adv}$, which is the debiased version of input image $\mathbf{x}$. By comparing the original reconstruction $\hat{\mathbf{x}}$ with $\hat{\mathbf{x}}_{adv}$, a practitioner should be able to recognize their difference, which corresponds to the bias. In Figure 4.3, we graphically display the latent adversarial perturbation.

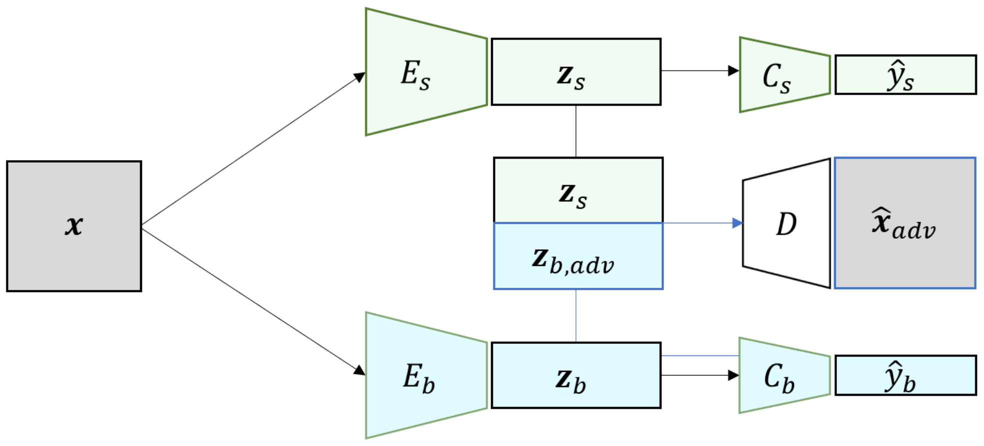


Figure 4.3: Graphical depiction of bias visualization through latent adversarial perturbation. In cyan are the discrepancies to the training structure in Figure 4.2. Through the latent adversarial perturbation, we generate the adversarially debiased representation $\mathbf{z}_{b,adv}$. This is then concatenated with $\mathbf{z}_s$ and passed through decoder D to visualize a debiased reconstruction $\hat{\mathbf{x}}_{adv}$.

While Deepfool (Moosavi-Dezfooli et al., 2016) works in the input space, we adapt it to operate in the latent space. The reason for the choice of perturbing in the latent space instead of the pixel space is that a slight change in bias can lead to a significant effect on the original input. For example, a small change in the position of an object in an image already has a big effect on the individual pixel values of said image. A minimal perturbation with respect to the input would then exploit some local imperfections of the classifier instead of correctly perturbing the bias in the image because such a change would be big in the pixel space.

An additional benefit of the latent adversarial perturbation utilizing the VAE is that they can be controlled to be inside the learned data distribution. Because of the probabilistic nature of the VAE, we have the prior distribution $p(\mathbf{z}_b)$, which allows us to detect outliers. Hence, we can require that $\mathbf{z}_{b,adv}$ is not an outlier to $p(\mathbf{z}_b)$ which creates more realistic perturbations. In our experiments, even without this regularization, $\mathbf{z}_{b,adv}$ stayed inside the data distribution, and thus we did not implement this feature.

We pick Deepfool over other adversarial attacks for two reasons: First, it tries to change the input as little as possible, whereas most other methods try to fool the classifier as much as possible within fixed perturbation boundaries. Using such methods would result in additional information being perturbed while for our visualization, we only want to remove the bias from an image and nothing else. Second, we want the perturbation to lie precisely on the decision boundary of two classes but not change it further. Otherwise, we not only debias the original representation but introduce a new bias from the new predicted class, which would complicate interpretation.

4.2.3 Implementation Details

An important part of this thesis is that we try to create a method that does not rely on architecture and hyperparameter tuning on specific data sets. Because in order to be applicable in a setting with unknown bias, we do not make the assumption that we have a bias-labeled validation set present for this purpose. Instead, we divide the space of possible data sets into two complementary subsets for which we provide separate architectures due to the inherent differences. As we worked exclusively with data sets consisting of images, the architectures are designed solely for this visual input type. However, extending to other data modalities should not be too difficult for future work because the changes are mainly in the encoder and decoder architecture which can be easily adapted to established VAEs in the alternative fields.

We divide images into uncomplicated, (semi-)synthetic and challenging, natural, realistic data sets. For example, all variants of MNIST (Lecun et al., 1998; Xiao et al., 2017; Kim et al., 2019) belong to the former, while all data sets consisting of photographs are part of the latter. We make this distinction because the difficulty in learning these data sets is inherently different and requires disparate treatment.

In line with previous works (Nam et al., 2020; Lee et al., 2021; Kim et al., 2021), we will use an MLP for learning synthetic data sets. The encoder consists of three linear layers with a bottleneck of size 100 for signal and bias, respectively. The decoder is again consisting of three linear layers. As activation function, we use the Rectified Linear Unit (ReLU). For the classifier, we solely use one linear layer for all data sets. This is because for us, the difficulty of a data set is determined by the complexity of the connection between latent variables and the realization $\mathbf{x}$ thereof. If we knew the latent variables, inferring the label would be simple. Thus, a single linear layer suffices.

We adapt the encoder-decoder structure for natural data sets from an MLP to a CNN, where we use a ResNet18 (He et al., 2016) for encoding and a ResNet18-like decoder. We decided against using a ResNet20, which Nam et al. (2020) use, on the basis that it requires average pooling of an 8×8 map for the bottleneck. In contrast to the pooling of a 2×2 map for ResNet18, this reduction is too strong of an information loss in a single layer such that it impedes good reconstructions. The distinction of data sets and respective models is necessary because for synthetic data sets, a vanilla CNN would learn bias and signal, thus making comparisons of methods impossible while also preventing visualizations of the bias that might be of interest to a practitioner.

For training our model, we utilize the final ELBO of Equation 4.4, which we repeat here for the sake of convenience.

$$\begin{aligned}\log[p(\mathbf{x}, y)] \geq \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[\log[p(\mathbf{x}|\mathbf{z}_s,\mathbf{z}_b)]\right] - D_{KL}\left[q(\mathbf{z}_s|\mathbf{x})q(\mathbf{z}_b|\mathbf{x})||p(\mathbf{z}_s)p(\mathbf{z}_b)\right] \\ + \mathbb{E}_{q(\mathbf{z}_s,\mathbf{z}_b|\mathbf{x})}\left[(1-\hat{y}_b)^q \log[p(y|\mathbf{z}_s)] + \hat{y}_b^q \log[p(y|\mathbf{z}_b)]\right]\end{aligned}$$

Note that the expectation over $q(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x})$ is replaced by a Monte-Carlo approximation usually consisting of a single draw of the latent variables. While this may seem like a bad approximation, NN are trained in batches where the loss is averaged over multiple samples, which in a stochastic sense is similar to drawing the latent variables multiple times. The ELBO needs to be maximized to make the lower bound on the likelihood of the data as high as possible. By convention, NN are trained by minimizing a loss function. For this reason, we negate the ELBO such that we can minimize it. To have an optimizable loss function, we need to make assumptions about the distributions contained in the ELBO.

We are working in a classification setting where y is either binary or categorical. Thus, we assume that $p(y|\cdot)$ corresponds to a categorical distribution for both classifiers. We will denote all possible classes that y can take on as y_j , $j = 1, \ldots, C$, where all $y_j = 0$, except for the correct label k for which $y_k = 1$. For notational consistency, we will denote the probability of each class j as predicted by our classifiers as $C(\mathbf{z})_j$. We omit the subscript implying bias or signal classifier as the following derivations hold for both. We have

$$\log[p(y|\mathbf{z})] = \sum_{j=1}^{C} y_j \log[p(y_j|\mathbf{z})] = \sum_{j=1}^{C} y_j \log[C(\mathbf{z})_j].$$

Recall that we negate the ELBO and thus its individual terms in order to minimize a loss function. With this we have recovered the well-known Cross Entropy (CE) Loss for the training of both classifiers:

$$-\log[p(y|\mathbf{z})] = -\sum_{j=1}^{C} y_j \log[C(\mathbf{z})_j] = CE$$

Next, we analyze the term corresponding to the Kullback–Leibler divergence (Kullback and Leibler, 1951). To get a closed form solution we assume for the priors that $p(\mathbf{z}_s)p(\mathbf{z}_b) = p(\mathbf{z}_s, \mathbf{z}_b) = \mathcal{N}(0, \mathbf{I}_{D\times D})$, where $\mathbf{I}_{D\times D}$ corresponds to the identity matrix of size D and D is the dimensionality of both latent representations together. Additionally, we assume that the approximate posteriors $q(\mathbf{z}_s|\mathbf{x})$ and $q(\mathbf{z}_b|\mathbf{x})$ take on independent Gaussian forms with diagonal covariance matrix $q(\mathbf{z}_s|\mathbf{x})q(\mathbf{z}_b|\mathbf{x}) = q(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I}_{D\times D})$, with $\boldsymbol{\mu}$ as mean vector and $\boldsymbol{\sigma}^2$ as variance vector of size D for the assumed-to-be independent latent representations. With these assumptions, we get

$$\begin{aligned} D_{KL}\big[q(\mathbf{z}_s|\mathbf{x})q(\mathbf{z}_b|\mathbf{x})||p(\mathbf{z}_s)p(\mathbf{z}_b)\big] &= D_{KL}\big[q(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x})||p(\mathbf{z}_s, \mathbf{z}_b)\big] \\ &= \iint q(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x}) \log\Big[\frac{q(\mathbf{z}_s, \mathbf{z}_b|\mathbf{x})}{p(\mathbf{z}_s, \mathbf{z}_b)}\Big] d\mathbf{z}_s d\mathbf{z}_b \\ &= -\frac{1}{2}\sum_{d=1}^{D} \big(1 + \log[(\sigma_d)^2] - (\mu_d)^2 - (\sigma_d)^2\big). \end{aligned}$$

For a detailed derivation, we refer to Odaibo (2019).

Lastly, we tackle the reconstruction term $\log[p(\mathbf{x}|\mathbf{z}_s, \mathbf{z}_b)]$. For (semi-)synthetic data sets, we assume that the channel-wise pixel values originate from a Bernoulli distribution. While this assumption might seem odd for RGB values, it helps account for the fact that most pixels are completely black, which corresponds to the RGB values (0,0,0). As the Bernoulli distribution is a special case of the Categorical distribution, we use the same reasoning to obtain the Binary Cross Entropy Loss (BCE):

$$\begin{aligned} -\log[p(\mathbf{x}|\mathbf{z}_s, \mathbf{z}_b)] &= -\sum_{l=1}^{L}\sum_{k=1}^{3} \log[p(x_{k,l}|\mathbf{z}_s, \mathbf{z}_b)] \\ &= -\sum_{l=1}^{L}\sum_{k=1}^{3}\sum_{j=1}^{2} x_{j,k,l} \log[D(\mathbf{z}_s, \mathbf{z}_b)_{j,k,l}] \\ &= \sum_{l=1}^{L}\sum_{k=1}^{3} BCE_{k,l} \end{aligned}$$

The indices j, k, l correspond to the Bernoulli realizations, three RGB channels, and the individual pixels, respectively.

On the other hand, natural, realistic data sets are more colorful. Thus, we assume that the RGB channels are independent and normally distributed with constant variance. We assume that all pixels are independent and identically distributed. Thus, the distribution of each pixel is described by $\mathcal{N}(\boldsymbol{\mu}, \sigma^2 \mathbf{I}_{3\times 3})$. Using this assumption we get

$$\begin{aligned}
-\log[p(\mathbf{x}|\mathbf{z}_s, \mathbf{z}_b)] &= -\sum_{l=1}^{L}\sum_{k=1}^{3} \log[p(x_{k,l}|\mathbf{z}_s, \mathbf{z}_b)] \\
&= -\sum_{l=1}^{L}\sum_{k=1}^{3} \log\Big[\frac{1}{\sigma\sqrt{2\pi}} \exp\Big(-\frac{1}{2}\Big(\frac{x_{k,l} - D(\mathbf{z}_s, \mathbf{z}_b)_{k,l}}{\sigma}\Big)^2\Big)\Big] \\
&= \sum_{l=1}^{L}\sum_{k=1}^{3} \log[\sigma\sqrt{2\pi}] + \frac{1}{2}\sum_{l=1}^{L}\sum_{k=1}^{3}\Big(\frac{x_{k,l} - D(\mathbf{z}_s, \mathbf{z}_b)_{k,l}}{\sigma}\Big)^2 \\
&= \sum_{l=1}^{L}\sum_{k=1}^{3} \log[\sigma\sqrt{2\pi}] + \frac{1}{2\sigma^2}\sum_{l=1}^{L}\sum_{k=1}^{3}(x_{k,l} - D(\mathbf{z}_s, \mathbf{z}_b)_{k,l})^2 \\
&\propto \frac{1}{2\sigma^2}\sum_{l=1}^{L}\sum_{k=1}^{3}(x_{k,l} - D(\mathbf{z}_s, \mathbf{z}_b)_{k,l})^2 \\
&= \frac{1}{2\sigma^2}\sum_{l=1}^{L} ||\mathbf{x}_l - D(\mathbf{z}_s, \mathbf{z}_b)_l||_2^2 \\
&= \frac{1}{2\sigma^2}\sum_{l=1}^{L} MSE_l,
\end{aligned}$$

where we have recovered a scaled pixel-wise Mean Squared Error (MSE). We omit the term that is constant with respect to the parameters of the decoder as it falls away when computing the derivative with respect to those weights.

For better training, we make two adaptions to the loss functions we derived. First, to have visualizations that capture the original image well, we upweigh the reconstruction loss by the factor 100, which corresponds to assuming that $\frac{1}{2\sigma^2} = 100$. Second, we rescale the reconstruction and KL term by dividing through $3L$ to be invariant to image resolution and number of channels while retaining their relative loss magnitude.

Finally, we have arrived at the final loss function when we insert all desiderata into the negated ELBO. For (semi-)synthetic data sets, we get

$$\begin{aligned}\mathcal{L} = \frac{1}{3L}\Big[&- 100 \sum_{l=1}^{L} \sum_{k=1}^{3} \sum_{j=1}^{2} x_{j,k,l} \log[D(\mathbf{z}_s, \mathbf{z}_b)_{j,k,l}] \\ &- \frac{1}{2} \sum_{d=1}^{D} \left(1 + \log[(\sigma_d)^2] - (\mu_d)^2 - (\sigma_d)^2\right)\Big] \\ &- (1 - \hat{y}_b)^q \sum_{j=1}^{C} y_j \log[C_s(\mathbf{z}_s)_j] - \hat{y}_b^q \sum_{j=1}^{C} y_j \log[C_b(\mathbf{z}_b)_j].\end{aligned}$$

For natural data sets, we get

$$\begin{aligned}\mathcal{L} = \frac{1}{3L}\Big[&100 \sum_{l=1}^{L} ||x_l - D(\mathbf{z}_s, \mathbf{z}_b)_l||_2^2 \\ &- \frac{1}{2} \sum_{d=1}^{D} \left(1 + \log[(\sigma_d)^2] - (\mu_d)^2 - (\sigma_d)^2\right)\Big] \\ &- (1 - \hat{y}_b)^q \sum_{j=1}^{C} y_j \log[C_s(\mathbf{z}_s)_j] - \hat{y}_b^q \sum_{j=1}^{C} y_j \log[C_b(\mathbf{z}_b)_j].\end{aligned}$$

This loss function is derivable with respect to the parameters of the modules such that we can update them. Note that we detach $\hat{y}_b$ from the computational graph as its purpose is solely the reweighting. For updating the model weights, we use the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.001 and batch size of 256 for all data sets. The hyperparameter q is chosen to be 0.7 by following the GCE coiners Zhang and Sabuncu (2018).

We perform early stopping and reduce the learning rate when plateauing by computing this loss function on a held-out 10% of the training set. For (semi-)synthetic data sets, we use an early stop patience of 2 versus 20 for realistic data sets. The patience for the learning rate reduction is one-half of the early stop patience and reduces the learning rate by a factor of 10.

For creating interpretable visualization of the bias, we adapt Deepfool (Moosavi-Dezfooli et al., 2016) for our purposes. Originally, it was developed for perturbing pixels in an input image, while we use it for perturbing latent dimensions. Thus, while pixel values need to be clamped in $[0, 1]$, we do not require this. For the distance measure of the perturbation, we use the ℓ_2-norm. Other popular options would be ℓ_{inf} or ℓ_1. We do not use the former because it would infer the assumption that bias is equally distributed among all latent dimensions. We decided against using the latter because the individual latent dimensions have no inherent meaning. For example, a rotation of 45° of the space does not matter for the latent space. Hence, it should not matter for the perturbation. While the ℓ_2-norm fulfills this criterion, the ℓ_1-norm does not.

For the bias visualization, we perturb images for which the biased and unbiased classifiers predict the correct class. With this, we aim to find bias-aligned images, which we can then perturb into neutral or bias-conflicting images. To differentiate the two, we can adapt the overshoot parameter, which is used for rescaling the optimal linearized perturbation. We usually keep the default of 102% but can make it bigger to create bias-conflicting images. One needs to keep in mind that a bigger overshoot might also start perturbing other aspects of the image. The choice of this hyperparameter is something that can be decided and adapted at inference time when looking at the visualizations.

Chapter 5

Experiments

In this chapter, we will present the setup we use to show the effectiveness of the method we developed in this thesis. In Section 5.1, we introduce the data sets on which we run our debiasing algorithm. Using a variety of data sets allows us to analyze the generalizability of all methods. To showcase the performance of our model, we compare it to state-of-the-art baselines, which we disclose in Section 5.2. Lastly, in Section 5.3, we report what metrics we use to measure the performance of the different methods.

5.1 Data Sets

In the following Subsections, we introduce the data sets on which we apply our method and the baselines. All data sets are designed to contain a majority of *bias-aligned* samples. In these images, each class coincides with a specific bias. Thus, for these bias-aligned images, a classifier can learn the bias instead of the signal and will still predict the correct class. However, in each data set, there is also a minority of *bias-conflicting* images. Leveraging the bias in these bias-conflicting images to predict the label leads to a wrong prediction. Therefore, to be generalizable, a debiasing algorithm should learn to leverage the signal instead of the bias. To analyze how well our method fulfills this goal, we evaluate and compare its performance.

First, in Subsection 5.1.1, we present a semi-synthetic data set with a synthetically generated bias for which a simple MLP is sufficient for learning. Second, in Subsection 5.1.2, we present a more challenging data set consisting of low-quality photographs, which were also induced with synthetic bias. Lastly, in Subsection 5.1.3, we tackle the running example showcased in Chapter 1.

5.1.1 Colored MNIST

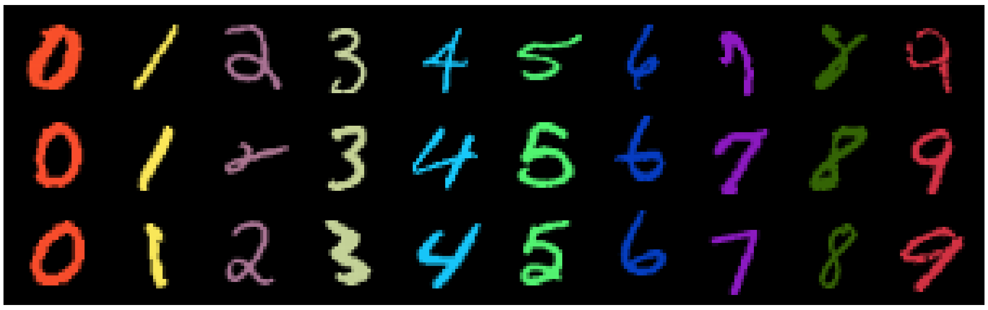


Figure 5.1: Bias-aligned images of the Colored MNIST data set. The columns show different digits $\mathbf{z}_s$ with their respective colors $\mathbf{z}_b$ that predominantly manifest in combination.

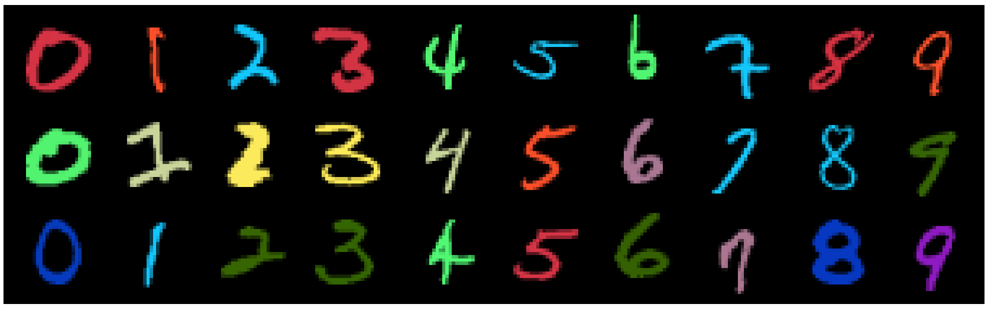


Figure 5.2: Bias-conflicting images of the Colored MNIST data set. The columns show different digits $\mathbf{z}_s$ with colors $\mathbf{z}_b$ that are usually not observed together.

Colored MNIST by Kim et al. (2019) is synthetically infused with a color bias to fulfill the graphical model from Subsection 4.1.1. Starting from the popular handwritten digit database MNIST (Lecun et al., 1998), the goal is to label the digit appearing in each 28×28 image. We randomly assign distinct mean colors to each digit that serve as bias attributes. Hence, the signal $\mathbf{z}_s$ is the digit while the easy-to-learn bias $\mathbf{z}_b$ manifests itself as the color.

According to the given percentage of bias-aligned images wanted, we sample for each image whether its digit will align with the corresponding color or if it will be conflicting. Then, the coloring of the digit is sampled from a normal distribution with a fixed mean of the specific color and standard deviation $\sigma = 0.005$. We do not use any preprocessing for this data set. In Figure 5.1, we show a selection of bias-aligned images for which digit and color match. Here, leveraging the color as decision rule would lead to the correct label. However, as this does not generalize well, we want to train a model that learns to recognize the digit instead of the color.

A minority of samples in the training set consists of bias-conflicting samples for which the biased decision rule leads to the wrong prediction. We show a random selection of bias-conflicting samples in Figure 5.2. Learning to recognize the digit instead of the color is the only valid decision rule with which bias-aligned as well as bias-conflicting samples can be correctly classified.

5.1.2 Corrupted CIFAR-10

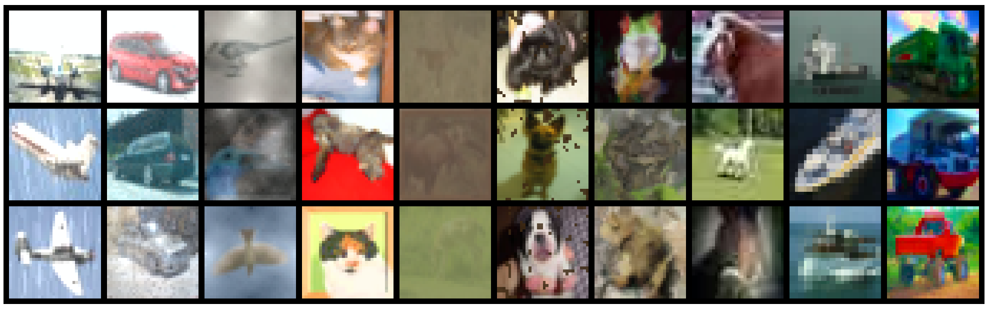


Figure 5.3: Bias-aligned images of the Corrupted CIFAR-10 data set. The columns show the different classes $\mathbf{z}_s$ with their respective corruptions $\mathbf{z}_b$ that predominantly manifest in combination. For example, the class birds often has foggy images, while ships are frequently pixelated.

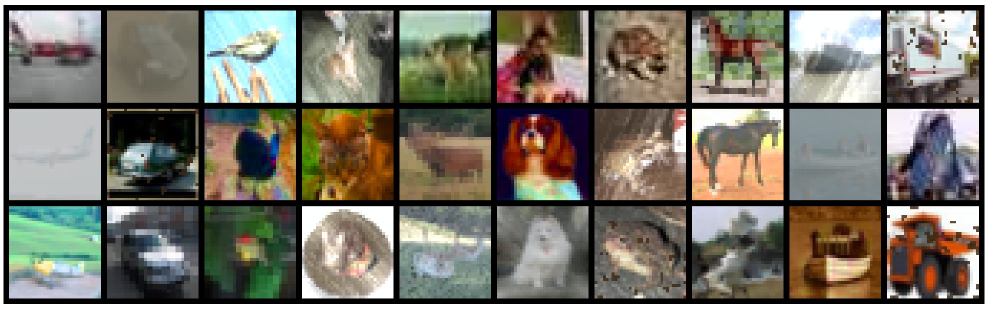


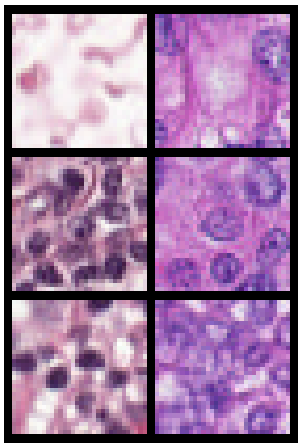
Figure 5.4: Bias-conflicting images of the Corrupted CIFAR-10 data set. The columns show the different classes $\mathbf{z}_s$ with corruptions $\mathbf{z}_b$ that are usually not observed together.

The first real-world data set we apply our method to is the Corrupted CIFAR-10 data set (Hendrycks and Dietterich, 2019). It is based on the standard CIFAR-10 data set (Krizhevsky and Hinton, 2009) that consists of images of different objects such as planes, ships, or dogs that an algorithm has to label. We then follow the protocol of Hendrycks and Dietterich (2019) and inject synthetically generated corruptions such as fog, brightness, or saturation for each class. These synthetic perturbations are designed to be as realistic as possible.

Identical to Colored MNIST, for each datum, we first sample whether it is bias-aligned or if it has a conflicting bias, according to the predefined probabilities. Afterward, we inject the perturbation into the image. The bias is then the corruption, while the signal is the object captured in the image. Some bias-aligned images of the data set can be found in Figure 5.3. The ten classes of Corrupted CIFAR-10 correspond to the ten columns in Figure 5.3 and from left to right are {`Airplane`, `Automobile`, `Bird`, `Cat`, `Deer`, `Dog`, `Frog`, `Horse`, `Ship`, `Truck`}. Similarly, the aligned biases from left to right are {`Snow`, `Frost`, `Fog`, `Brightness`, `Contrast`, `Spatter`, `Elastic`, `JPEG`, `Pixelate`, `Saturate`}. On top of that, in Figure 5.4, we depict bias-conflicting images, which contain combinations of signal and bias that are rarely observed in the training set.

For preprocessing, we take random crops consisting of at least 50% of the original image and resize it to the original 32×32 size. This leads to more diversity, which prevents overfitting on the data. Additionally, we allow horizontal flips of the images and standardize the pixel values over the entire data set. To calculate the reconstruction loss, we transform the standardized pixel values back into $[0, 1]$ so that its size is comparable among all data sets.

5.1.3 Camelyon17


(a) Bias-aligned images

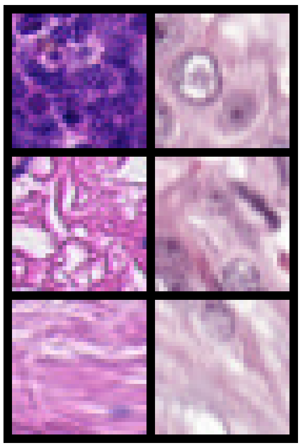

(b) Bias-conflicting images

Figure 5.5: Images of the Camelyon17 data set. The columns of the subfigures correspond to the label. In the first column are images without tumor, while in the second column are tumorous images. For bias-aligned images, an image with a tumor originates from hospital A, while images without a tumor come from hospital B. In bias-conflicting images, the opposite is true. The biasing discrepancy of the hospitals manifests in differently colored patches.

While both previous data sets contain synthetically injected biases, we use Camelyon17 (Bándi et al., 2019; Koh et al., 2021) to create a biased training set without the need for changing the individual images. Additionally, this medical data set offers the application of our method in a setting where interpretability is incredibly important and unobserved biases can lead to detrimental effects. The data set consists of histopathological whole-slide images from five dutch hospitals for which the goal is to determine whether a tumor is present or not. The five medical centers are Radboud University Medical Center in Nijmegen (RUMC), Canisius-Wilhelmina Hospital in Nijmegen (CWZ), University Medical Center Utrecht (UMCU), Rijnstate Hospital in Arnhem (RST), and the Laboratory of Pathology East-Netherlands in Hengelo (LPON).

We start off from the patch-based variant by Koh et al. (2021) of the Camelyon17 data set by Bándi et al. (2019). The 96×96 patches have labels that correspond to whether a tumor is present in the center 32×32 patch. For our purposes, we Center Crop the images to single out this patch. Originally, the test set in Koh et al. (2021) has been chosen to consist of all images from UMCU to induce a distribution shift. This specific hospital was chosen because it is visually most distinctive from the hospital images contained in the training set. For our purposes, we want to leverage this difference to induce a bias.

To create our training set, we combine RUMC from the original training set with the visually disparate UMCU. Then, we sample mostly tumourous images from the latter while doing the opposite for the former. As test set, we sample 1250 images for each class from CWZ, RST, and LPON, which were not included in the training set generation. Thus, we have recovered the example introduced in Chapter 1. We present an assortment of bias-aligned patches in Figure 5.5.

Here, the bias is the hospital, while the signal is the presence of a tumor. Using the hospital from which the patch originated as decision rule only leads to correct labels in the bias-aligned images of the training set. Similar to Corrupted CIFAR-10, we allow horizontal flipping and randomly crop and resize the image to have more diverse images. This risks that some training images will not contain a tumor but still be labeled positive due to the cropping. We are aware of this risk and believe that the overfitting-reducing effect of the cropping outweighs its inducing bias.

5.2 Baselines

In order to evaluate the performance of our method, we have determined three baselines to which we compare the proposed approach. We use the same encoder and classifier as in our method for all baselines to have comparable results. This means we encode synthetic data sets using a three-layered MLP with a bottleneck dimension of 100 for bias and signal, respectively. For more challenging data sets, we use a ResNet18 (He et al., 2016) with a respective bottleneck dimension of 512 as the encoder. The classifiers are always a single linear layer. For each method, we apply the preprocessing specified in the subsections of Section 5.1.

The first baseline we implement is a Vanilla model, which measures the standard performance without any debiasing scheme. This method only uses one classifier instead of two. We train it for 100 or 200 epochs for synthetic and real-life data sets, respectively. Analyzing the performance of a non-debiasing method helps us assess whether the required assumptions of a biased setting are fulfilled and shows whether the debiasing algorithms have a positive effect at all.

The second model we compare the proposed approach to is LfF from Nam et al. (2020). We use their work as a comparison because they have pioneered debiasing without explicit bias labels. Lastly, we compare our method to DisEnt by Lee et al. (2021), a recently proposed state-of-the-art debiasing algorithm that showed promising results.

As an additional benefit, DisEnt offers visualizations of the bias, which means that we can compare our method to theirs quantitatively as well as qualitatively using reconstructed images. As the baseline papers have not been tuned on Camelyon17, we adopt the hyperparameters and model from their Corrupted CIFAR-10 architecture to simulate the absence of an unbiased validation set.

5.3 Evaluation

In order to be able to fairly and reproducibly compare the performance of our method to the baselines on the various data sets, we first define the metrics used to evaluate performance. We measure the debiasing capabilities of the algorithms by calculating their accuracy on an unbiased test set. In this test set, bias and signal are independent and uniformly distributed, so there is an equal amount of data samples for each bias and signal combination. Thus, for the synthetically generated Colored MNIST and Corrupted CIFAR-10, we generate equally many samples for each combination of bias and signal for the test set. A biased classifier, which learned the biased decision rule due to the vast amount of bias-aligned images in the training set, will perform poorly for this unbiased test set as the spurious correlation between the bias and the label does not exist anymore.

For Camelyon17, we evaluate the performance of the algorithms by calculating the accuracy on the three hospitals not included in the test set, where for each hospital, we sample the same amount of positive and negative samples for a total of 7500 data points. We use the numbers zero to nine as different random seeds for generating all the data sets and initializing the models' weights at the start of training. We do this to reduce the variability and cherry-picking potential of the evaluation.

To further investigate the behavior of the different algorithms, we vary the percentage of bias-conflicting samples in the training set. For each data set, we calculate the performance where the ratio of bias-conflicting samples to the entire training set is $\{0.005, 0.01, 0.02, 0.05, 0.1, 0.2\}$. The higher this number is, the more samples are in the training set, for which leveraging the bias as decision rule does not lead to the correct class prediction. This variation allows us to investigate how the different methods adapt when the difficulty of learning the bias changes in a given data set.

As we have pointed out in the previous chapter, a significant advantage of our method is the interpretable visualization of the bias. While this is not measurable quantitatively, we will perform a qualitative analysis. For this, we will compare the bias visualizations offered by DisEnt with our method. We generate a collection of five image reconstructions simultaneously to show the performance variability in a data set. For both methods, we select a case where they perform well while also including the visualization of the other method. Additionally, in the Appendix, we depict randomly selected visualizations for all data sets. Vanilla and LfF do not offer such a feature. Thus, they are excluded from this part of the analysis.

For our visualization of the bias, we single out images for which the signal and bias classifiers predict the correct label. We expect that these images are bias-aligned. Next, we sample a target class different from the true label. Then, using our latent adversarial perturbation, we perturb $\mathbf{z}_b$ such that the biased classifier predicts this wrong class. This procedure removes the bias from the representation such that a classifier that leverages the bias predicts the wrong label. Thus, by this perturbation, we have removed the bias from the bias-aligned representation, which we then visualize by reconstructing both images. The change in both images should correspond to the bias that this class is associated with. For example, for a bias-aligned image of Colored MNIST, we would expect our perturbation to perturb the red color of the digit 0 while keeping the shape of the digit fixed.

To visualize the bias, DisEnt (Lee et al., 2021) swaps the bias representation of two images to break the connection between the signal and bias in bias-aligned images. When generating their bias visualization, it is crucial that we perform their approach using identical encoders, decoder, and classifiers as our method. This enables us to compare both bias visualization strategies directly. If we used the trained model of DisEnt for applying their bias visualization, it would be ambiguous whether differences between our method and DisEnt's arise due to the different bias visualization approaches or due to the different models and representations. As we want to compare the bias visualization strategies separately from the training, at inference time, we will use the representations and model weights trained by our method and apply DisEnt's swapping approach.

As a starting point for DisEnt's swapping approach, we utilize the same supposedly bias-aligned images used in our method. Then, we randomly sample images for which the bias classifier predicts the specific target class used in our perturbation. Therefore, the sampled images' bias conflicts with the original images' signal. Thus, we replace $\mathbf{z}_b$ of the original images with the sampled images' latent bias dimensions to obtain a bias-conflicting representation. Consequently, by reconstructing the original supposedly bias-aligned image together with its adapted bias-conflicting version, we would expect to observe the bias of the corresponding class.

Chapter 6

Results

In the following section, we will present the results of our experiments on the different data sets. In Section 6.1, we describe how we report the results of all models. Then, in Section 6.2, Section 6.3, and Section 6.4, we present the results of our experiments on Colored MNIST, Corrupted CIFAR-10, and Camelyon17, respectively.

6.1 Presentation of Results

For quantitatively evaluating the debiasing capabilities of each model, we report the average accuracy on the unbiased test set as well as the estimated standard deviation of a single run. For each setting, we compare the best performing model with the second best through a two-sided paired t-test to determine whether they differ significantly. This test assumes that the underlying proportions are approximately normally distributed. We argue that this is the case as the accuracy is calculated as an average over all random seeds and test samples for which we can invoke the central limit theorem. If the null hypothesis of equality can be rejected on a 5% significance level, then we indicate this by **bolding** the accuracy. If the null hypothesis can not be rejected, we <u>underline</u> all methods that are not significantly worse than the one with the highest empirical accuracy.

For qualitatively evaluating the bias visualization capabilities, we show the reconstructions with Ours' and DisEnt's visualization strategies using our trained model as a backbone for both. We first show the original image with its corresponding reconstructed image. Then, we visualize the debiased representations according to Ours' and DisEnt's method. We cherry-pick two collection of images, one for each work, where the methods show good performance. Additionally, in the appendix, we visualize a randomly selected assortment of images.

6.2 Colored MNIST

Colored MNIST (Kim et al., 2019) is the only (semi-)synthetic data set of our experiments. Most of the model design was performed on this data set. Due to the nature of the setting, there is no validation set with labels for the bias. Thus, model architecture design and hyperparameter tuning have been performed by considering test set performance. This holds true for both baseline papers (Nam et al., 2020; Lee et al., 2021) as well as our method. The interpretation of this section's results should consider this fact.

In Table 6.1, we show the performance of all models on the unbiased test set of Colored MNIST. We see that Vanilla significantly outperforms the debiasing algorithms for the 10% and 20% cases. The debiasing methods show their benefit only for a lower amount of bias-conflicting samples in the training sets. Here, our method seems to outperform or at least match all baselines while DisEnt always is the runner-up. Especially for the 0.5% setting, there appears to be a considerable gap in performance between our and other methods. In the 2%, 1%, and 0.5% environments, all debiasing methods outperform the vanilla model, which indicates their usefulness. The estimates differ from the values presented in the baseline papers (Nam et al., 2020; Lee et al., 2021) because we also vary random seeds over data set generation instead of only over the model.

Colored MNIST	Baselines			
Bias-conflicting	Vanilla	LfF	DisEnt	Ours
20%	**94.92** ± 0.24	70.18 ± 4.19	90.94 ± 1.46	85.24 ± 1.60
10%	**91.24** ± 0.26	81.99 ± 5.01	89.12 ± 1.44	85.35 ± 1.23
5%	<u>85.48</u> ± 0.50	81.18 ± 2.94	<u>85.54</u> ± 2.49	<u>86.14</u> ± 1.78
2%	73.28 ± 0.56	76.97 ± 2.49	<u>82.38</u> ± 1.68	<u>83.80</u> ± 1.28
1%	59.41 ± 0.39	68.91 ± 5.01	76.33 ± 3.41	**80.03** ± 2.04
0.5%	43.70 ± 0.83	60.42 ± 2.72	63.98 ± 4.78	**71.63** ± 2.49

Table 6.1: Unbiased accuracy + standard deviation in % for Colored MNIST. For each percentage of bias-conflicting samples in the training set, the method with the significantly highest accuracy is denoted in **bold**. If no method is significantly better than the others, all methods not significantly worse than the highest are <u>underlined</u>.

In Figure 6.1, we picture the bias visualization from DisEnt and our method. Both collections of images are specifically selected to show the potential of either approach, as described in Section 6.1. In Figure B.1 of Appendix B, we additionally present a collection of randomly selected images.

In general, DisEnt perturbs the bias representations so strongly that this also leads to a change in the digit. This is because the bias and signal representations are not perfectly disentangled. Thus, the leftover signal in the bias dimensions gets swapped. On the other hand, our method does not perturb the digit while regularly perturbing the color. However, due to the weaker magnitude of change, our approach sometimes does not visibly change the image at all.

A similar impression arises when analyzing the cherry-picked Figure 6.1. We see that the visualizations of our method are always close to the original reconstruction, while DisEnt shows bigger changes in signal as well as bias. For the left collection, we notice that our method successfully changes the color of the reconstruction while keeping the signal. The degree of change varies in the images. Conversely, DisEnt changes the color as well as making the digit unreadable. On the other hand, in the collection on the right, DisEnt manages to nicely perturb only the color, while our method for some images does not visibly change the reconstructions at all. Notably, our approach also nicely perturbs the bias for the second, fourth, and fifth columns.

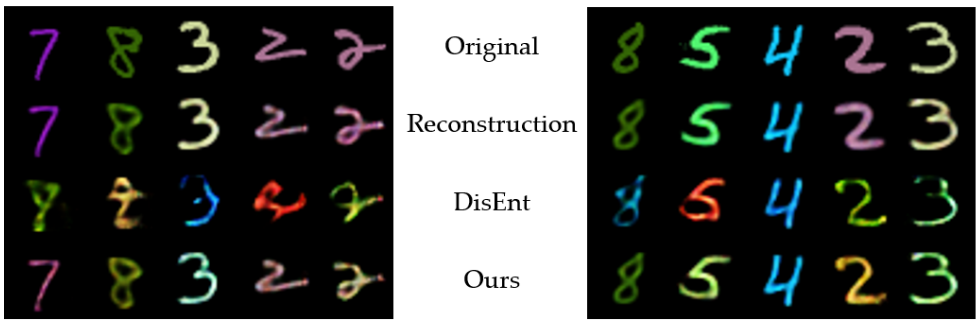


Figure 6.1: Selected visualizations of bias for Colored MNIST. Both collections of images are cherry-picked for presenting the potential of the respective visualization method. On the left side, we depict images for which our method shows promising results, while on the right side are visualizations working well for DisEnt.

6.3 Corrupted CIFAR-10

Similar to the previous data set, due to the absence of an unbiased validation set, it is to be expected that the baselines tuned their hyperparameter to this data set and individual settings by peeking at the test set performance. As for us, we have used this data set to design the general architecture for all natural data sets while keeping the training method we developed with Colored MNIST.

In Table 6.2, we show the performance of all models on the unbiased test set of Corrupted CIFAR-10. The best performing models are LfF and Ours. We observe that for higher percentages of bias-conflicting samples, our method is better than LfF, while for lower proportions, the opposite is the case. DisEnt seems to be generally worse than the other debiasing methods. However, it makes an interesting peak in performance for the 0.5% case, which might be attributed to randomness as it does not follow the usual trend of decreasing performance with decreasing bias-conflicting samples. Finally, Vanilla performs worse than the debiasing methods except for the 20% case. Thus, the debiasing methods present an improvement over a standard empirical risk minimizer.

Corrupted CIFAR-10	Baselines			
Bias-conflicting	Vanilla	LfF	DisEnt	Ours
20%	<u>67.57</u> ± 0.41	64.50 ± 2.17	60.99 ± 5.84	<u>66.75</u> ± 1.34
10%	57.11 ± 0.76	<u>59.29</u> ± 3.16	53.47 ± 4.43	<u>61.26</u> ± 2.06
5%	46.89 ± 0.78	<u>55.77</u> ± 2.33	46.40 ± 5.81	<u>55.63</u> ± 1.54
2%	34.90 ± 0.81	<u>47.26</u> ± 1.56	36.98 ± 4.43	<u>43.66</u> ± 1.81
1%	28.22 ± 0.73	**39.39** ± 2.16	31.22 ± 2.69	35.17 ± 1.19
0.5%	22.26 ± 1.03	<u>30.04</u> ± 1.67	<u>31.97</u> ± 3.34	27.30 ± 2.04

Table 6.2: Unbiased accuracy + standard deviation in % for Corrupted CIFAR-10. For each percentage of bias-conflicting samples in the training set, the method with the significantly highest accuracy is denoted in **bold**. If no method is significantly better than the others, all methods not significantly worse than the highest are <u>underlined</u>.

In Figure 6.2, we picture the bias visualization from DisEnt and our method. On the left, the selected collection of images for our approach shows that we managed to remove the saturation bias for the trucks in the first and fourth columns. It is not possible to recognize the JPEG and pixelation bias in the second and third columns. In the last column, the visualization of our method has become much darker, indicating a bias related to brightness, which with further research, might lead a practitioner to the actual bias that is snow. DisEnt does not work well for these images, as the swapping of the bias vectors also perturbs the image's content. Arguably, for the last column, DisEnt shows that the bias from the swapped image is contrast because nothing is visible anymore.

The right collection of images in Figure 6.2 is meant to show visualizations where DisEnt performs well. It was challenging to find pictures where their approach did not change the image's content entirely. Here, in the first column, the visualization of DisEnt has much more contrast than the original reconstruction, which indicates a bias. Also, the saturation in the trucks in the third and fifth columns has changed to be brighter, even if the content also changed a bit. As for our method, one can see small changes in saturation for the third image that make the colors look less extreme and thus show a bias. For the other images, the bias visualizations are relatively similar to the original reconstructions.

In Figure B.2 of Appendix B, we additionally present a collection of randomly selected images. There, one can observe the general behavior that for most visualizations, our method does not change the image visibly, while DisEnt changes the complete content of the picture. This recurrent behavior for Corrupted CIFAR-10 shows that both approaches rely on the underlying model's disentangling capabilities, which should also be considered when attempting to improve bias visualizations.

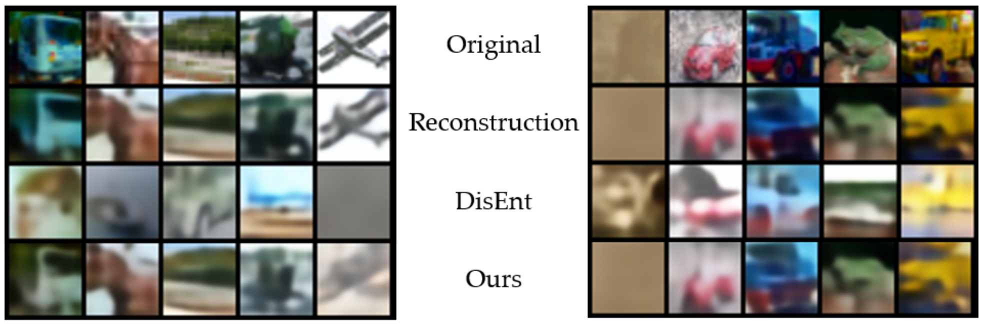


Figure 6.2: Selected visualizations of bias for Corrupted CIFAR-10. Both collections of images are cherry-picked for presenting the potential of the respective visualization method. On the left side, we depict images for which our method shows promising results, while on the right side are visualizations working well for DisEnt.

6.4 Camelyon17

The binary data set Camelyon17 (Koh et al., 2021) has been curated by us to contain a bias corresponding to different hospitals. We did not adapt any architectural choices or perform hyperparameter tuning on this data set to have a perfectly unbiased estimate and comparison of performance among all models.

In Table 6.3, we depict the performance of all methods. Notably, Vanilla consistently performs better than or equal to the debiasing methods. When analyzing only those, we see that our method outperforms or matches the performance of the other two debiasing algorithms. Of those two, LfF shows better results than DisEnt. Interestingly, there is great variability of the runs in the lower half of bias-conflicting percentages for all debiasing methods.

In Figure 6.3, we show visualizations selected for demonstrating the debiasing capabilities of DisEnt and our method. In the left collection, the first column shows an image containing a tumor. The tumor's presence correlates with the picture being from another hospital, which manifests itself as a lilac color. Our method accurately changes the lilac color while keeping the tumor content in the image. When comparing to DisEnt, we observe that their method perturbs the image strongly such that other information from the

Camelyon17	Baselines			
Bias-conflicting	Vanilla	LfF	DisEnt	Ours
20%	**65.72** ± 2.36	31.98 ± 1.76	31.40 ± 2.19	45.52 ± 3.71
10%	**65.02** ± 1.67	37.87 ± 2.87	35.34 ± 2.44	48.62 ± 4.82
5%	**64.05** ± 2.98	42.05 ± 10.07	39.71 ± 4.13	53.57 ± 5.92
2%	<u>62.77</u> ± 2.47	49.46 ± 7.83	44.28 ± 3.55	<u>65.22</u> ± 4.58
1%	<u>61.44</u> ± 1.53	<u>60.14</u> ± 8.74	53.44 ± 5.80	<u>64.42</u> ± 4.95
0.5%	<u>60.85</u> ± 1.81	<u>65.65</u> ± 7.45	56.15 ± 6.41	<u>59.16</u> ± 7.83

Table 6.3: Unbiased accuracy + standard deviation in % for Camelyon17. For each percentage of bias-conflicting samples in the training set, the method with the significantly highest accuracy is denoted in **bold**. If no method is significantly better than the others, all methods not significantly worse than the highest are <u>underlined</u>.

image is lost, additionally to the change in color. For the non-tumorous pictures in the left collection, our method changes the color visibly in the second, fourth, and fifth columns, even though for the second column, it is barely visible in comparison to the intense color change of DisEnt. For all four non-tumorous images, DisEnt adequately swaps the color corresponding to the other hospital. Although, their method also generates some artifacts, for example, adding and removing some cell nuclei in the third column.

In the right collection of Figure 6.3, DisEnt very elegantly colors the non-tumorous images of the first three columns in lilac. At the same time, it removes said color in the two other images that contain the tumor. This happens without creating too many artifacts, except for slight changes in the second column. Our method correctly perturbs the second and fourth images while leaving the other images visibly unchanged.

Additionally, in Figure B.3 of Appendix B, we present a collection of randomly selected images. In these visualizations, one can observe that generally, our bias visualization method tends to color all reconstructions in lilac. Images that already are in this color stay unchanged. In comparison, DisEnt regularly changes colors in both directions. Due to its strong perturbation, it can visualize the bias in most images but often creates artifacts in the process. In the next chapter, we will argue that the softer approach of our method is preferable for a practitioner.

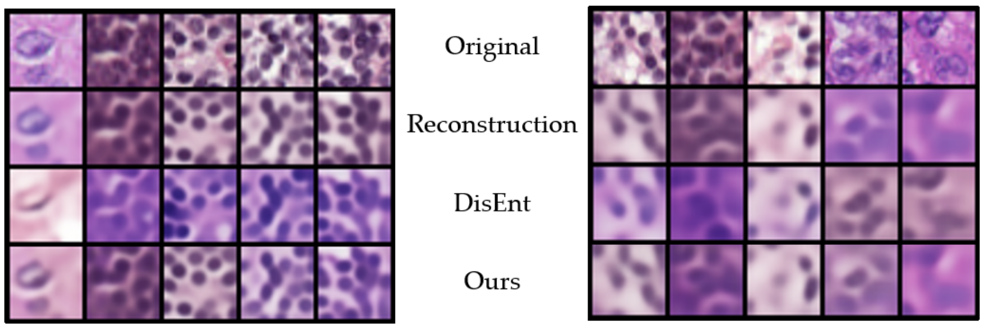


Figure 6.3: Selected visualizations of bias for Camelyon17. Both collections of images are cherry-picked for presenting the potential of the respective visualization method. On the left side, we depict images for which our method shows promising results, while on the right side are visualizations working well for DisEnt.

Chapter 7

Discussion

In this chapter, we discuss the improvements of our method compared to previous work by interpreting the results of our experiments. First, in Section 7.1, we review the reliance on a bias-labeled validation set of all debiasing methods. Next, in Section 7.2, we examine the baseline methods' debiasing capabilities and show how our method's adjustments pose an advantage over previous works. In the following Section 7.3, we extend the accuracy-based analysis by the advocated aspect of visualizing the bias and discuss the differences in DisEnt and our method. Lastly, in Section 7.4, we debate the interpretability of our approach.

7.1 Absence of Validation Set

There cannot be a bias-labeled validation set in a setting of unknown, hidden bias. Thus, debiasing methods for such data sets must not rely on such a set for optimizing their architecture and tuning their hyperparameters. Of all debiasing methods without individual bias labels, to the best of our knowledge, all works apart from JTT (Liu et al., 2021) completely ignore this problem. Thus, we want to call for more transparency, honesty, and critical discussions regarding this subject.

So far, all debiasing methods for hidden bias have relied on the presence of a validation set where the bias was labeled for each data point. As has been argued in previous chapters, this is counter-intuitive to the underlying idea of this setting where the bias is unknown. This thesis presents a method that does not require hyperparameter tuning. By theoretically founding our loss function in the ELBO, we remove the need for weighting each element of the loss, as the ELBO predetermines the relative size thereof. This removes the need to assume the presence of a bias-labeled validation set for our method.

Furthermore, to the best of our knowledge, we are the first method in the hidden bias domain that uses a biased validation set, where we split the initial training data into training and validation sets. Both of which do not contain labels on the bias. By integrating the VAE with its ELBO, we can perform early stopping on this biased validation set and reduce the learning rate when plateauing. In contrast to previous methods, this works because by including a generative model in the architecture, we can train until this model has learned the data distribution.

To be transparent, we want to state that we have used Colored MNIST and Corrupted CIFAR-10 for designing our training scheme and model architecture for (semi-)synthetic and realistic data sets, respectively. As there is no bias-labeled validation set, we used their test sets for evaluating performance gains, always considering the average over ten random seeds. Therefore, the results of our method on these two data sets are biased as it is possible that we overfitted on the data distribution. Regarding the debiasing baselines LfF (Nam et al., 2020) and DisEnt (Lee et al., 2021), it is reasonable to expect that in their works, they optimized their architecture and hyperparameters on those test sets as well. As Vanilla also makes use of our architectural choices, we argue that the comparison of all methods is still fair. The reason being that all methods possibly overfitted to the data distributions of Colored MNIST and Corrupted CIFAR-10 to some degree. To get a completely unbiased evaluation of performance, we introduced Camelyon17, for which no method was allowed to adapt anything.

7.2 Quantitative Analysis

In this section, we will analyze the debiasing capabilities of our as well as the baseline methods. The accuracy estimates we obtained differ from the values presented in the baseline papers (Nam et al., 2020; Lee et al., 2021). This is because we also vary random seeds over data set generation instead of only over the model parameters. This is also why our experiments' estimated standard deviations are higher than those stated in the baselines. We advocate for including data set generation when varying the random seeds, as this reduces the potential of architectural overfitting. Especially in our setting of hidden bias where architectures are designed by looking at the test set performance, this at least prevents overfitting on a specific test set.

When interpreting the results, we evaluate the performance of the debiasing methods only for the settings where Vanilla did not perform best. This is because, for the cases where a standard empirical risk minimizer learns the signal and outperforms debiasing methods, it is safe to conclude that the easy-to-learn assumption is not fulfilled. Generally, it is coherent that Vanilla performs better the more bias-conflicting samples are contained in the training set. Because an increase in bias-conflicting data points weakens Subassumption 2c, which states that the easiness of learning outweighs the number of bias-conflicting samples for which the biased decision rule does not work. Thus, weakening the debiasing methods while strengthening the vanilla model.

Overall, for settings where the easy-to-learn assumption is likely to be fulfilled, our method shows auspicious results. In 11/13 of those cases, we are better or equal than the other debiasing methods. We take this as proof that our adaption from reweighting the samples for the signal classifier by the RDS in LfF and DisEnt, to reweighting by $(1 - \hat{y}_b)^q$ is not only a simplification, but also improves performance notably.

One critical point with the RDS is that its calculation requires the use of an exponential moving average on its individual cross entropy terms for training stability. This is problematic because it unintentionally hinders the downweighting of bias-aligned samples for the signal classifier if the bias is learned quickly by the biased classifier.

The reason for this is that in the first training epoch, both classifiers will not be able to predict the label well. Thus, we expect the RDS to be around 50% for each sample. As soon as the biased classifier has picked up on the biased decision rule, the RDS should become very small for bias-aligned data points. For Colored MNIST, this learning of the bias usually happens in the first epoch. However, due to the exponential moving average on the individual terms of the RDS, the weighting of the next epoch takes into account that before, the RDS weighting was 50%. Using LfF's decay parameter of 0.7, in the optimal case, only after 12 epochs will the weight be at 1%. This means that until then, in a 1% bias-conflicting setting, a signal classifier would put more aggregated weight on bias-aligned samples than others. Therefore, it will learn the biased decision rule first, before attempting to learn the true, unbiased decision rule. We argue that unlearning an existing decision rule is more challenging than learning the correct decision rule from scratch. For this reason, our reweighting strategy by $(1 - \hat{y}_b)^q$, which does not require an exponential moving average, improves the performance. Lastly, the RDS also uses a class-wise max-normalization whose function is unclear while considerably influencing performance. Contrary to that, our reweighting is explicit in what it is doing.

We want to emphasize the performance of our approach on Camelyon17, the only data set on which no method was able to tune their architecture or hyperparameters. In this binary data set, our method is the best of all considered debiasing methods. We hypothesize this is the case because LfF's and DisEnt's signal classifiers also learn the bias. Theoretically, they train solely on bias-conflicting data points, which for a binary data set, makes them biased too. This is because if a well-performing decision rule for bias-aligned samples is that hospital A implies no tumor, then for bias-conflicting data points, the opposite decision rule of hospital A implies tumorous will work. Thus, there is no need for either classifier to learn the signal. This can not happen with our method because the VAE requires that the signal, which is necessary for reconstruction, has to be learned in the latent dimensions.

Interestingly, for some settings in this binary data set, LfF and DisEnt perform worse than random guessing. We hypothesize that their training time is too long due to not having early stopping. We believe that by training for too long, their biased classifier learns the signal, similar to Vanilla, and gives the unbiased classifier samples to learn for which this correct decision rule leads to the wrong solution. For example, images where only a tiny part of it is tumorous. Thus, the signal classifier learns the opposite decision rule, which will not work for all samples on which the bias classifier, which learned the signal decision rule, performs well. In a nutshell, we hypothesize that for them, both classifiers usually predict the opposite class given an image, where the biased classifier uses the correct decision rule. This theory is supported by the fact that the accuracy of vanilla is roughly one minus the accuracy of LfF and DisEnt for the settings where a biased classifier would learn the signal.

The findings on Camelyon17 indicate that our method, which is specifically designed not to require data set specific hyperparameter tuning, is generalizable for data sets where there is no bias-labeled validation set, while the other methods are not.

Over all data sets, our method has a lower variability in performance than the other debiasing methods, which we attribute to the fact that we are able to use learning rate reducing measures. Those require using a biased validation set during training, which only our method does. The high variability in Camelyon17 for all methods implies that the signal classifier only finds the signal in some cases. Thus, future work could be conducted to make this more stable such that the signal is detected more regularly.

Overall, these results show that our debiasing method can match or exceed the performance of state-of-the-art debiasing methods while not relying on hyperparameter tuning.

7.3 Qualitative Analysis

Now, we will compare the bias visualizing capabilities of DisEnt and our method. Recall that in a setting of hidden bias, it is beneficial for the practitioner to obtain information about possible spurious correlations contained in a data set.

In the figures of Chapter 6 as well as Appendix B, it is clearly visible that the swapping of DisEnt perturbs the images stronger than our method. This is to be expected, as they substitute the whole latent bias dimensions while our method tries to perturb this representation as little as possible. While in DisEnt, the true bias is emphasized more, their images also contain artifacts due to the strong perturbation. For example, in Figure B.3, DisEnt adds or removes cell nuclei quite frequently. Conversely, our method either perturbs the bias or does not perceivably change the reconstruction.

We will now argue why our conservative visualizations are more desirable. For real-life data sets, whose images consist of more than solely signal and bias, the VAE will have to store this content somewhere in the latent dimensions. As this information is irrelevant to the label, it is plausible that some of the image content will be learned in the bias dimensions. By swapping the bias dimensions of two images, this image-specific content gets swapped too, leading to non-similar reconstructions. Then, a practitioner can not unambiguously observe the bias when comparing original reconstruction and swapped bias visualization because content apart from the bias has been swapped too. Our method does not have this downfall, as our goal is to perturb the bias as little as possible, which results in all label-independent information staying constant while only the bias changes.

The presence of information in an image outside of signal and bias is also why we claim that our training of the decoder simultaneously with the encoders and classifiers is to be preferred. This is opposed to DisEnt, which train their decoder only after having trained everything else. Their encoders are trained by minimizing the loss of the classifiers. We believe that for real-life data sets, if the decoder is trained with fixed encoders that focus on only learning bias and signal, some label-independent information in the images gets lost. Thus, training the decoder only after fixing the encoders and classifiers makes reconstruction a more challenging task and possibly leads to less precise visualizations. Therefore, training the encoders and the decoder separately hinders visualization capabilities regarding the bias.

An additional reason why we argue that our method is to be preferred is that from the viewpoint of a practitioner that does not know the bias a priori, all artifacts of DisEnt have to be considered as possible bias. While for our method, not having a visible change in a reconstruction does not pose a potential bias. With our method, a practitioner can skim through images until they see a change, which they can then attribute to being the bias. The bias does not need to be visible in every image. Using DisEnt's method, on the other hand, a practitioner would have to consider every artifact as a possible bias which is not helpful. This difference is especially visible for cases where signal and bias are not perfectly disentangled, such as in Figure B.2.

Here, in Corrupted CIFAR-10, the visualizations of DisEnt heavily change the image's content, while our method often does not visibly change the reconstruction at all. This implies that the learned latent representations are not learned or disentangled well. Because by swapping the bias dimensions, the content also changes. Additionally, the bias classifier also appears not to have meaningful decision boundaries as seemingly every point is close to a border. In this entangled latent space, we observe that our method is more lenient towards not perfectly disentangled representations. As our visualizations at least keep the content of the original image. While so far, we have used a standard VAE and disentangled the latent representations by leveraging the differently weighted signal and bias classifiers, future work could investigate the application of more sophisticated, disentangling VAEs.

If we managed to disentangle signal and bias perfectly, then it would be possible to train the signal classifier without downweighting bias-aligned samples. However, trying this showed that perfect disentanglement has not yet been achieved, as such an unweighted training decreased performance. Regardless, we postulate that our model is more designed towards this goal than DisEnt. Because their classifiers take bias as well as signal dimensions as their input. If the representations were truly disentangled, then the classifiers should not see the other representation. But then, their swapping would also not work anymore.

Regarding the overall lower performance in Corrupted CIFAR-10 for all methods, we theorize that in this data set, not every bias is easier to learn than the signal. For example, in Figure 6.2, the JPEG bias of horses or the elastic perturbations bias from frogs for a human seem more challenging to recognize than the animal. Additionally, we theorize that a limitation of our current VAE is that it cannot recognize all biases in Corrupted CIFAR-10. For example, in Figure 6.2, the plain reconstructions of the car do not show the frost bias of the original image. Thus, it can not be expected that the biased classifier picks up on an attribute that the encoder is not sophisticated enough to capture. Note that this statement does not contradict the easy-to-learn assumption because it concerns the limitations of the model architecture and not the actual content of the image itself. For example, if a model were designed only to see black and white, then it would not be able to recognize the color bias in Colored MNIST, while the easy-to-learn assumption still clearly holds. Therefore, future work could investigate not only disentangling VAEs but also generative models, such as the NVAE (Vahdat and Kautz, 2020), which are able to capture more details of the images in their latent space and reconstructions.

In Figure 6.1, in the left collection, the images containing the digit 2 have different strengths of color perturbation for our method. This nicely shows that depending on the target class of the perturbation, we get different visualizations. For the digit two in the fourth column, the target bias is red, which is closer to pink, and thus the bias visualization shows less of a change. While for the fifth column, the target color's decision boundary is farther away, therefore, the change is more prominent. This finding indicates that a practitioner can also experiment with choosing different target classes of the latent adversarial perturbation for the same image to see which attributes change regularly. This can help in deducing the hidden bias.

While these visualizations have to goal of helping a practitioner to deduce the hidden bias, they also have an auxiliary purpose. If the assumptions of our model are not fulfilled, then this might be determined by analyzing the visualizations. For example, if the easy-to-learn assumption is not fulfilled, then the bias classifier will learn to detect the signal instead of the bias. Therefore, in the visualizations, the signal instead of the bias of the image would be changing. This would indicate to a practitioner that the bias classifier learned the signal instead of the bias. In this case, there is no need for a debiasing algorithm, and applying a standard empirical risk minimizer such as Vanilla should be preferred. By having interpretable visualizations of what the model believes to be bias, we allow for the inspection of the assumptions required for our method.

7.4 Interpretability

Initially, the goal of this thesis was to use the latent adversarial perturbation during training and not only at inference time. Unfortunately, doing this considerably decreased performance, so we discarded the idea. The reason for the reduction in accuracy is that the debiasing perturbations are not consistent enough, as can be observed in the randomly selected figures of Appendix B. They are not consistent enough because the latent representations are not perfectly disentangled and because adversarial attacks tend to exploit local imperfections of a classifier's decision boundary instead of truly perturbing the bias.

Thus, we leverage our latent adversarial perturbation solely for bias visualization at inference time. Here, not every visualization needs to be perfectly debiased, as a practitioner can recognize biases even if only a subset of visualizations show it.

Nonetheless, we believe it is correct to call our method interpretable. This is because the decoder trained for visualizing the bias is trained simultaneously with the classifiers and not after. Hence, it is an inherent part of the model instead of a post hoc analysis tool. We adopt the distinction of Lipton (2018), who states that interpretability asks how the model works while explainability asks what else the model can tell. As the decoder is an inherent part of the model, the perturbations visualized by it are a part of how the model works. Nevertheless, we acknowledge that arguably the latent adversarial perturbation itself falls into the category of explainable methods because it is not used during training. Thus, our bias visualizing method is a bit of both.

Training an interpretable model is in contrast to the methods of the baseline papers. Nevertheless, we show promising results when compared to their performance. This indicates that we are able to overcome the well-known accuracy-interpretability trade-off as we show good performance against non-interpretable baselines while being able to interpretably visualize hidden biases.

Chapter 8

Conclusion

In this thesis, we have tackled the problem of hidden bias. In this setting, a classifier leverages spurious correlations to make its predictions. Without proper treatment of this situation, the application of such an algorithm can have adverse effects in the real world, where such a spurious correlation does not hold anymore.

This thesis advances the research in creating an unbiased classifier that does not fall prey to such biases. For defining the environment in which our debiasing method is expected to unfold its potential, we define the graphical model and state the assumptions that give rise to a setting where an empirical risk minimizer would fall prey to spurious correlations.

For training an unbiased classifier, we derive a novel reweighting scheme based on our theoretical considerations and train a generative model with it. In this model, we train a bias classifier to be as biased as possible and simultaneously train an unbiased classifier by upweighting samples for which the biased decision rule does not work well. We show that our simple weighting factor can match or outperform existing works. Additionally, by training a generative model, we are able to generate interpretable visualizations of the bias at inference time. For this, we leverage latent adversarial perturbations, with which we improve upon existing works as we do not introduce artifacts.

We advocate for more transparency and collaborations toward the tuning of hyperparameters, as future methods should focus on training in the absence of a bias-labeled validation set. Furthermore, we believe our work can be extended by improving the VAE structure so that it disentangles and has better capabilities for capturing and reconstructing biases.

Overall, we show that our interpretable method can overcome the accuracy-interpretability trade-off. We show good performance against non-interpretable baselines while being able to interpretably visualize hidden biases.

Appendix A

Evidence Lower Bound

$$
\begin{aligned}
\log[p(\mathbf{x})] &= \int q(\mathbf{z}|\mathbf{x})\log[p(\mathbf{x})]d\mathbf{z} \\
&= \int q(\mathbf{z}|\mathbf{x})\log\Big[\frac{p(\mathbf{x},\mathbf{z})}{p(\mathbf{z}|\mathbf{x})}\frac{q(\mathbf{z}|\mathbf{x})}{q(\mathbf{z}|\mathbf{x})}\Big]d\mathbf{z} \\
&= \int q(\mathbf{z}|\mathbf{x})\log\Big[\frac{q(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})}\Big]d\mathbf{z} + \int q(\mathbf{z}|\mathbf{x})\log\Big[\frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})}\Big]d\mathbf{z} \\
\log[p(\mathbf{x})] &\geq \int q(\mathbf{z}|\mathbf{x})\log\Big[\frac{p(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})}\Big]d\mathbf{z} \\
&= \int q(\mathbf{z}|\mathbf{x})\log\Big[\frac{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{q(\mathbf{z}|\mathbf{x})}\Big]d\mathbf{z} \\
&= \int q(\mathbf{z}|\mathbf{x})\log[p(\mathbf{x}|\mathbf{z})]d\mathbf{z} - \int q(\mathbf{z}|\mathbf{x})\log\Big[\frac{q(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})}\Big]d\mathbf{z} \\
&= \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\big[\log[p(\mathbf{x}|\mathbf{z})]\big] - D_{KL}\big[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\big]
\end{aligned}
$$

Appendix B

Bias Visualizations

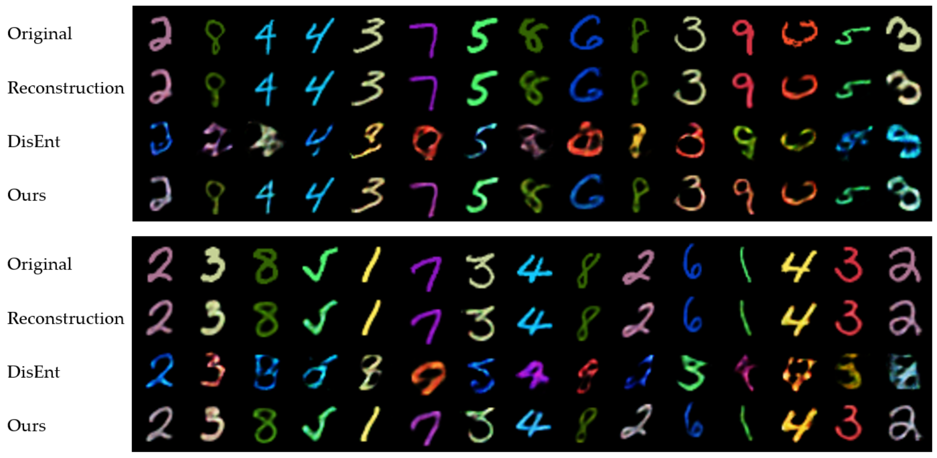


Figure B.1: A random collection of bias visualizations for Colored MNIST. The randomly selected images are varied over random seeds and the percentage of bias-conflicting images in the training set.

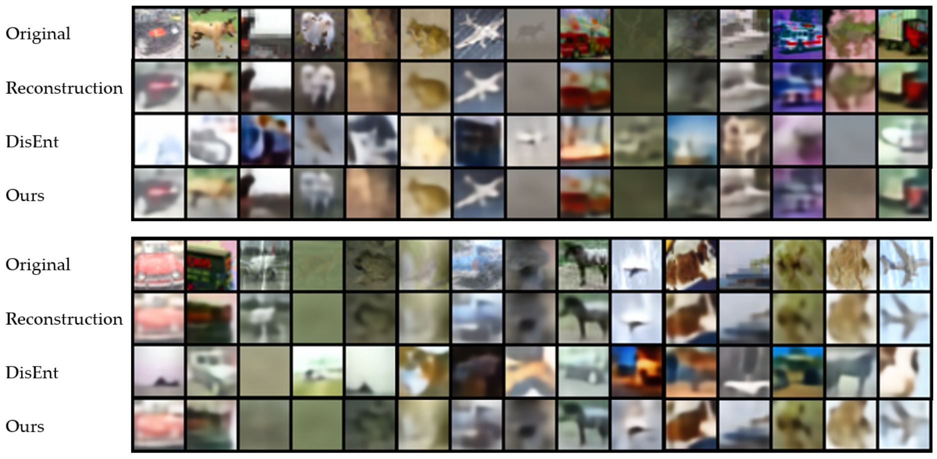


Figure B.2: A random collection of bias visualizations for Corrupted CIFAR-10. The randomly selected images are varied over random seeds and the percentage of bias-conflicting images in the training set.

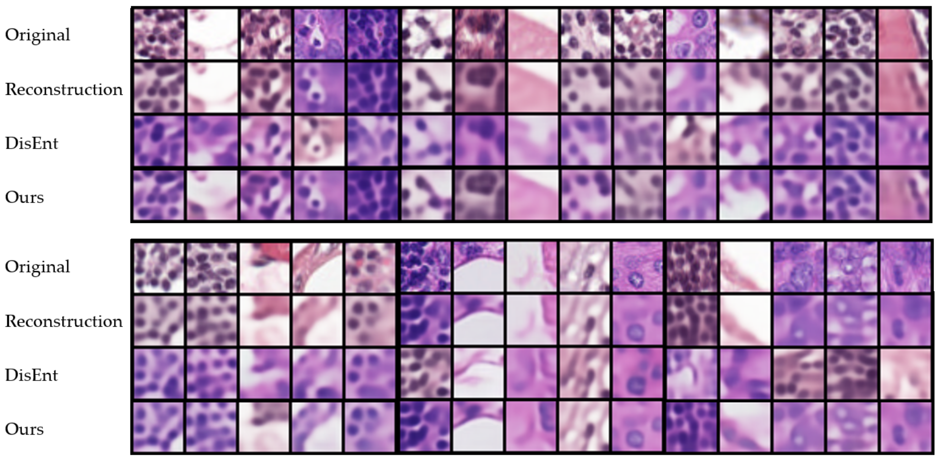


Figure B.3: A random collection of bias visualizations for Camelyon17. The randomly selected images are varied over random seeds and the percentage of bias-conflicting images in the training set.

Bibliography

Angwin, J., Larson, J., Mattu, S., and Kirchner, L. (2016). Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing. [Online; Retrieved 18 July 2022].

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. (2017). A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR.

Bahng, H., Chun, S., Yun, S., Choo, J., and Oh, S. J. (2020). Learning de-biased representations with biased representations. In *International Conference on Machine Learning*, pages 528–539. PMLR.

Bándi, P., Geessink, O., Manson, Q., Dijk, M. V., Balkenhol, M., Hermsen, M., Bejnordi, B. E., Lee, B., Paeng, K., Zhong, A., Li, Q., Zanjani, F. G., Zinger, S., Fukuta, K., Komura, D., Ovtcharov, V., Cheng, S., Zeng, S., Thagaard, J., Dahl, A. B., Lin, H., Chen, H., Jacobsson, L., Hedlund, M., Çetin, M., Halici, E., Jackson, H., Chen, R., Both, F., Franke, J., Küsters-Vandevelde, H., Vreuls, W., Bult, P., van Ginneken, B., van der Laak, J., and Litjens, G. (2019). From detection of individual metastases to classification of lymph node status at the patient level: The CAMELYON17 challenge. *IEEE Trans. Medical Imaging*, 38(2):550–560.

Beutel, A., Chen, J., Zhao, Z., and Chi, E. H. (2017). Data decisions and theoretical implications when adversarially learning fair representations. *arXiv preprint arXiv:1707.00075*.

Box, G. (1979). Robustness in the strategy of scientific model building. In LAUNER, R. L. and WILKINSON, G. N., editors, *Robustness in Statistics*, pages 201–236. Academic Press.

Clark, C., Yatskar, M., and Zettlemoyer, L. (2019). Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 4067–4080.

Creager, E., Madras, D., Jacobsen, J.-H., Weis, M., Swersky, K., Pitassi, T., and Zemel, R. (2019). Flexibly fair representation learning by disentanglement. In *International conference on machine learning*, pages 1436–1445.

Darlow, L., Jastrzebski, S., and Storkey, A. (2020). Latent adversarial debiasing: Mitigating collider bias in deep neural networks. *arXiv preprint arXiv:2011.11486*.

Dressel, J. and Farid, H. (2018). The accuracy, fairness, and limits of predicting recidivism. *Science advances*, 4(1):eaao5580.

Edwards, H. and Storkey, A. (2016). Censoring representations with an adversary. In *International Conference on Learning Representations*.

Flores, A. W., Bechtel, K., and Lowenkamp, C. (2016). False positives, false negatives, and false analyses: A rejoinder to "machine bias: There's software used across the country to predict future criminals. and it's biased against blacks.". *Federal probation*, 80.

Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F. A., and Brendel, W. (2019). Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*.

Hardt, M., Price, E., and Srebro, N. (2016). Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems*, pages 3315–3323.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778. IEEE Computer Society.

Hendrycks, D. and Dietterich, T. G. (2019). Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*. OpenReview.net.

Kearns, M., Neel, S., Roth, A., and Wu, Z. S. (2018). Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In *International Conference on Machine Learning*, pages 2564–2572. PMLR.

Kim, B., Kim, H., Kim, K., Kim, S., and Kim, J. (2019). Learning not to learn: Training deep neural networks with biased data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9012–9020.

Kim, E., Lee, J., and Choo, J. (2021). Biaswap: Removing dataset bias with bias-tailored swapping augmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14992–15001.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. (2014). Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In *International Conference on Learning Representations*.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., Lee, T., David, E., Stavness, I., Guo, W., Earnshaw, B., Haque, I., Beery, S. M., Leskovec, J., Kundaje, A., Pierson, E., Levine, S., Finn, C., and Liang, P. (2021). WILDS: A benchmark of in-the-wild distribution shifts. In Meila, M. and Zhang, T., editors, *International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5637–5664. PMLR.

Krizhevsky, A. and Hinton, G. (2009). Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto, Ontario.

Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, J., Kim, E., Lee, J., Lee, J., and Choo, J. (2021). Learning debiased representation via disentangled feature augmentation. In *Advances in Neural Information Processing Systems*.

Li, Y. and Vasconcelos, N. (2019). Repair: Removing representation bias by dataset resampling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9572–9581.

Lin, T., Goyal, P., Girshick, R. B., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In *IEEE International Conference on Computer Vision*, pages 2999–3007. IEEE Computer Society.

Lipton, Z. C. (2018). The mythos of model interpretability. *ACM Queue*, 16(3):30.

Liu, E. Z., Haghgoo, B., Chen, A. S., Raghunathan, A., Koh, P. W., Sagawa, S., Liang, P., and Finn, C. (2021). Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582.

Nam, J., Cha, H., Ahn, S., Lee, J., and Shin, J. (2020). Learning from failure: De-biasing classifier from biased classifier. In *Advances in Neural Information Processing Systems*.

Namkoong, H. and Duchi, J. C. (2016). Stochastic gradient methods for distributionally robust optimization with f-divergences. In *Advances in Neural Information Processing Systems*, pages 2208–2216.

Odaibo, S. G. (2019). Tutorial: Deriving the standard variational autoencoder (VAE) loss function. *CoRR*, abs/1907.08956.

Park, T., Zhu, J., Wang, O., Lu, J., Shechtman, E., Efros, A. A., and Zhang, R. (2020). Swapping autoencoder for deep image manipulation. In *Advances in Neural Information Processing Systems*.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. (2020). Distributionally robust neural networks. In *International Conference on Learning Representations*. OpenReview.net.

Vahdat, A. and Kautz, J. (2020). NVAE: A deep hierarchical variational autoencoder. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*.

van den Oord, A., Vinyals, O., and Kavukcuoglu, K. (2017). Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pages 6306–6315.

Wang, H., He, Z., Lipton, Z. C., and Xing, E. P. (2019). Learning robust representations by projecting superficial statistics out. In *International Conference on Learning Representations*.

Weinzaepfel, P. and Rogez, G. (2021). Mimetics: Towards understanding human actions out of context. *International Journal of Computer Vision*, 129(5):1675–1690.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747.

Zhang, B. H., Lemoine, B., and Mitchell, M. (2018). Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 335–340.

Zhang, Z. and Sabuncu, M. R. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in Neural Information Processing Systems*, pages 8792–8802.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2016). Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Interpretable Approach to Discover and Remove Hidden Biases

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Vandenhirtz	Moritz

With my signature I confirm that
- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Zürich, 05.09.2022	[illegible]

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.