# Invariance in Neural Representations with Applications in Fairness and Debiasing

Mattia Cerrato

October 2020

# Contents

# Chapter 1

# Introduction

It has been a tumultuous decade for Machine Learning and Artificial Intelligence. Connectionist approaches under the name of "deep neural networks" have seen a resurgence, estabilishing state of the art results in a plethora of applications. The most well-known result is perhaps due to Alex Krizhevsky et al. [KSH12] who showed how Yann LeCunn's convolutional networks [LeC+90] could be made deeper (i.e. with many hidden layers) and faster to train via GPU programming. The authors' model, dubbed "AlexNet", pushed the object recognition performance on the ImageNet dataset [Den+09] by more than 10 percentage points in terms of top-1 accuracy. Since then, deep neural networks have been dominating the ImageNet leaderboards on object recognition and detection. Neural networks are now the de facto standard for many computer vision applications.

Neural networks have also been employed in recent years in speech-to-text applications. Since Graves' Connectionist Temporal Classification in 2014, other models based on Recurrent Neural Networks have provided impressive benefits in terms of WER (Word Error Rate). Success stories in Natural Language Processing are numerous, but the most impressive and suggestive are perhaps in language modelling, where the GPT family of models is able to generate human-passing text.

In short, Machine Learning models based on neural networks are now part of our everyday life, being deployed in smartphones and embedded systems. However, neural networks are not without their issues. While the new wave of connectionist systems displays impressive performance in various tasks, these models are very hard to understand in various ways. Neural network training objectives are highly non-convex, which impedes convergence guarantees. While in practice this does not seem to be an issue, it makes it hard to understand training progress. Furthermore, there is a tension between classic machine learning theory, which analyzes a model's generalization performance via complexity measures such as its VC dimension, and neural networks: some ResNet models [He+16] have a number of trainable parameters in the tens of millions. The usual connectionist rebuttal here underlines the number of neurons in the brain, which is estimated to be around 100 billion; however, the recently developed GPT-3 language model employs 175 billion learnable weights. It goes without saying that these models are not at all human readable: no symbolic computation is involved, and in case of incidents it is nigh impossible to understand what has gone wrong.

The technical optimism and excitement around Machine Learning has also pushed businesses to apply it in situations where it may impact people's well-being directly, such as loan applications [Ols11], candidate selection for job offers [CLM20; BR18] and evaluating the chance of re-offending for people who commited crimes [Ang+16]. Computer vision applications based on neural networks have even been employed to judge beauty contests [Lev16]. In such contexts, opaque models are particularly problematic as there is a concrete risk for discrimination against certain groups of people. Protected characteristics such as gender and ethnicity might be used in the computation of the final decision, which is problematic on moral grounds and unlawful. These pieces of information are usually called *sensitive attributes* in the fair ML literature. While it is of course possible to exclude these attributes from the training data, opting for what has been called a “colorblind” approach (by e.g. Zehlike et al. [ZC19]), this is often not sufficient to avoid discrimination. The existence of the gender pay gap [WW05], for example, shows how there are complex correlations between sensitive attributes and non-sensitive attributes such as a person’s yearly salary. If models are learning from biased data, it follows that they will learn to output biased decisions; if we are unable to explain those decisions, we are left with very little human control over what ultimately is a software process. On top of being philosophically troubling and unethical, recent legislation might see these methodologies as unlawful. Taking the General Data Protection Regulation in the European Union as an example, transparency and fairness have to be guaranteed to individuals who are subject to automatic decision-making software systems.

In absence of fully explainable models, the fair ML literature has focused on *non-discriminating* models, under many possible definitions. The approach here is usually to reformulate training objectives so that they can include non-discrimination (fairness), but pre-processing and post-processing approaches have also been developed (see e.g. the contribution by Kamiran and Calders [KC09]). In this context, neural networks provide both challenges and opportunities. On one hand, new methodologies and training objectives have to be developed so to constrain neural networks for fairness; on the other, there is opportunity to learn *invariant representations*. To explain this concept, let us track back to the gender pay gap example, where there is an obvious but complex correlation between the gender of the individual and their salary. If it were possible to train a neural network to actively remove that correlation in its internal representations - the neuronal activations - then we would have fairness guarantees even in an overall non-explainable model. The overarching approach of this thesis is to explore this idea and develop Machine Learning methodologies which can obtain invariant representations.

In the next chapter I will explore the concept of "invariance" in statistics with a focus on Mutual Information. Chapter 3 goes deeper into the intuitive ideas of non-discrimination and fairness. Chapter 4 introduces the relevant connectionist literature. My own experimental work is contained in Chapter 5, drawing from papers I have published during the last three years. My contributions might be summarized as follows:

1. **A noise module for invariant representation learning**. Invariant representation learning requires ad-hoc architectures on top of fairness-focused learning objectives. Inspired from recent academic discussion on the *equitability* of Mutual Information, I developed a new layer which can be included in any neural network architecture. This contribution has emerged from a collaboration with Dr. Laura Li Puma.

2. **A fair pairwise ranker**. An extension of a state of the art pairwise ranker so that it is constrained for fairness. The noise module again proves to be beneficial, both in the developed architecture and in others already present in the literature. This body of work has been developed during a visiting period in Prof. Dr. Stefan Kramer's group at the Johannes-Gutenberg Universität in Mainz, Germany.

3. **An interpretable fairness framework**. Via a simple architectural change, it is possible to constrain neural architectures to learn *fair*

*corrections* instead of fair representations. This manages to somewhat open the "black box" of deep networks and guarantee an explanation.

4. **Fair Group-Shared Representations**. Leveraging invertible neural network models, I show how to learn a mapping between different groups of individuals. The result is that individuals from different groups are "mixed together" in feature space and indistinguishable in practice.

The last three topics have seen contributions by Dr. Alexander Segner and Dr. Marius Köppel from the University of Mainz. All four would not have been possible without the contributions and guidance of my advisor, Prof. Roberto Esposito.

# Chapter 2

# Invariance

The concepts of invariance and independence is often used in data analysis and statistics to establish that two events or variables have no relation between them, that is, obtaining information about one of the two does not give any information about the other. Two random variables are independent iff $p(X, Y) = p(X)p(Y)$. Proving that two random variables are independent requires deep knowledge of the data generating process. In other situations, it may be sensible to assume independence; when it is not, one might try to *quantify* the level of dependence (or correlation) between two variables. There are a plethora of available measures to achieve this goal, the most well-known being the Pearson correlation coefficient. However, this measure is limited as it is only able to capture linear correlations. Other measures, which are also able to detect non-linear relationships between two or more variables, are the Mutual Information and the Maximum Mean Discrepancy (developed by Gretton et al. [Gre+12]). While these metrics are reasonably cheap and simple to compute when having full information about the distributions involved, estimating their value from a data sample is challenging. In our setting, which involves statistical models that only represent a distribution *implicitly*, it is paramount to employ efficient, unbiased estimators for the aforementioned measures. In the following, I shall discuss the concepts of invariance and independence and review a number of different measures of dependence and correlation, while commenting on their applicability to the purpose of evaluating neural representations.

## 2.1 Pearson Correlation Coefficient

The Pearson Correlation Coefficient is widely employed to obtain information about the *linear* relationships between two random variables. It is defined as:

$$\rho_{X,Y} = \frac{E[(X - E[X])(Y - E[Y])]}{\sigma_x \sigma_y} = \frac{Cov(X,Y)}{\sigma_x \sigma_y} \tag{2.1}$$

Where we used $E$ as notation for the expected value, $Cov(X,Y)$ for the covariance and $\sigma_x$ for the standard deviation of $X$. The Pearson Correlation Coefficient is defined between -1 and 1, and since if $p(X,Y) = p(X)p(Y)$ then $cov(X,Y) = 0$, it follows that $p(X,Y) = p(X)p(Y) \rightarrow \rho_{X,Y} = 0$. However, the Pearson Correlation Coefficient cannot be employed to test independence at large, since the inverse is in general not true [Mur12]. For instance, if $X \sim Uniform(-1,1)$ and $Y \sim X^2$, $Cov(X,Y) = 0$ even though one variable is a function of the other.

Because of its relationship with the linear regression problem, where the slope of the regression line can be computed by $cov(X,Y)/var(X)$, Pearson's Correlation Coefficient is often thought of as a measure of linear correlation between two variables. This limits its applicability for our purpose of evaluating the invariance of neural representations w.r.t. some nuisance variable. As modern neural architectures employ a number of non-linear activation functions (see Chapter 4), more complex correlation measures which can account for non-linear relationships have been preferred by many authors [Gre+12; TZ15; Yu+20; Gan+16].

## 2.2 Mutual Information

The Mutual Information measure has been introduced in the context of noisy channel coding by Shannon [Sha48]. It can be defined for both discrete and continuous random variables:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y)\ log \frac{p(x,y)}{p(x)p(y)} \tag{2.2}$$

$$I(X;Y) = \int_X \int_Y p(x,y)\ log \frac{p(x,y)}{p(x)p(y)} dx\ dy \tag{2.3}$$

It is easy to see that if $X$ and $Y$ are independent, the Mutual Information is 0. To build an intuition about the Mutual Information, it is relevant to see its connection with the Kullback-Leibler (KL) Divergence. The KL Divergence is defined between two probability mass functions $p(X)$ and $q(X)$ as follows:

$$D_{KL}(p \mid\mid q) = \sum_{x \in X} p(x)\ log\ \frac{p(x)}{q(x)} \tag{2.4}$$

Or, equivalently, when $p$ and $q$ are densities defined over a continuous random variable:

$$D_{KL}(p \mid\mid q) = \int_{-\infty}^{+\infty} p(x)\ log\ \frac{p(x)}{q(x)}\ dx \tag{2.5}$$

In general, the KL divergence measures the distance between the two distributions, with the important caveat that it is not a metric as it does not respect the triangle inequality [Mur12]. Furthermore, the KL divergence can be defined in terms of the cross entropy $H(p, q)$:

$$D_{KL}(p \,||\, q) = -H(p, p) + H(p, q) \tag{2.6}$$

$$H(p, q) = -\sum_{x \in X} p(x) \; log \; q(x) \tag{2.7}$$

$$H(p, p) = -\sum_{x \in X} p(x) \; log \; p(x) \tag{2.8}$$

It is possible to show that the cross entropy is the average number of bits one needs to encode data coming from distribution $p$ when using $q$ to define a codebook[1] [CT06]. The measure is not symmetric as, in general, it is not true that this value is equal when one instead uses $p$ to define a codebook for $q$. In contrast, the entropy $H(p, p)$ is the expected number of bits one needs to define a codebook if using the true distribution $p$; therefore, the KL divergence is the difference between these two values. Besides this intuitive result, it is possible to show formally [Mur12; CT06] that the KL divergence is positive. Thus, it is possible to interpret Mutual Information as the KL divergence between two distributions $p(X, Y)$ and $p(X)p(Y)$ by substituting them in Equation 2.2:

$$I(X; Y) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \; log \frac{p(x, y)}{p(x)p(y)} \tag{2.9}$$

$$= D_{KL}(p(X)p(Y) \,||\, p(X, Y)) \tag{2.10}$$

Mutual Information is an attractive measure of independence as it is 0 only when $p(x, y) = p(x)p(y)$. However, it requires knowledge about the joint distribution which can be expensive or impossible to acquire. Furthermore, many neural network models express a distribution only implicitly, not to mention that the encoded distribution for discriminative models is usually

[1]A codebook is a mapping from values of $p$ to symbols which can then be transmitted over a noisy channel.

the conditional $p(y \mid x)$ where $y$ is the ground truth and $x$ is the input data. In our case we are however interested in evaluating the neural representations $x^l$ at some level $l$ w.r.t. some nuisance factor $s$. It is relevant to note that $p(x^l \mid s)$ is not usually included in the training objective of discriminative networks, whereas $p(y \mid x)$ is usually optimized for via the maximum likelihood criterion [Mur12] by means of a cross entropy loss. In this situation, we are limited to sampling from $p(x^l)$. It is also challenging to estimate the joint distribution $p(x^l, s)$ with sampled data only. However, the properties of Mutual Information are attractive enough that many authors have focused on developing *estimators* for it, both in a general sense and more specifically to analyze the "Information Flow" inside a deep neural network (see e.g. [TZ15]). In the following, I propose a review of these topics and discuss their applicability to our setting. In Subsection 2.3.3, we will see how it is also possible to develop upper bounds for the mutual information and include the term $p(x^l \mid s)$ in a loss function.

## 2.3 Estimating Mutual Information

As briefly discussed above, estimating the mutual information between two random variables (or, as in our setting, vectors) is a challenging task. Detecting non-linear relationships between variates is of interest in many fields. As a matter of fact, this problem has attracted contributions from scientists in various disciplines, from pure statistics to signal processing and machine learning. In the following, I will review some of the approaches with a focus on more recent literature.

### 2.3.1 Histogram-based Approaches

As touched upon previously, estimating Mutual Information is tightly tied to estimating the density of the involved probability distributions. Per Krasov et al., [KSG04] the most straightforward approach is to employ a "binned approximation" as follows[2]:

$$I(X;Y) \approx I_{bin}(X;Y) = \sum_{i,j} p_{bin}(i,j) log \, \frac{p_{bin}(i,j)}{p^x_{bin}(i) p^y_{bin}(j)}$$

Where $p^x_{bin}(i)$ is obtained by dividing the domain of $X$ into a finite number of bins and calculating how many samples fall into bin $i$, then dividing by the sample size $N$. Similarly, iterating over index $j$, for $p^y_{bin}(j)$. Lastly, $p_{bin}(i,j)$ is computed by counting how many samples fall at the intersection of the $i$-th and $j$-th bin. Intuitively, this family of methods is equivalent to building a histogram of the distributions from a sample. This simple discretization has the benefit of converging to the actual value of $I(X;Y)$ when $N \to \infty$ and when the bins become infinitely small [CT06]. This methodology has been referred to as the "direct" [Pan03] or "plug-in" method and has been employed in different applications, including Independent Component Analysis [Alm03] and neuroscience [Str+98]. One thing to note is that the bin sizes need not be fixed. The Maximal Information Coefficient (MIC) of Reshef et al. [Res+11] is one among the methods which search for "optimal" bin sizes. For each pair of integers $(m, n)$ in a defined range, this methodology considers a number of possible grids (i.e. bin sizes, without restriction on equality over $X$ and $Y$). $I(X;Y)$ is then estimated on the obtained discrete distributions as defined above. The grid displaying the highest value for the estimation is the computed MIC value. Reshef et al. also include a

[2]nota per me: una discussione formalizzata si puo' trovare in gretton05

normalization term which ensures that the maximum MIC is 1. While MI is not upper bounded, the MIC still satisfies the more relevant property to our purpose: i.e. that it is 0 iff $X$ and $Y$ are independent.

### 2.3.2 Kernel Approaches

Just as histogram-based approaches may be employed in density estimation and therefore in mutual information estimation, the same is true for kernel methods. In kernel density estimation, the hyperrectangle-sized bins of the preceding section are replaced with kernel functions. One kernel function is placed at each sample's position. Each sample's contribution to the density estimate is then computed following the kernel's shape (usually a Gaussian). Let $x_1 ... \, x_n$ be a sample from the random variable (or vector) $X$. Then, the kernel density estimate can be defined as follows:

$$\hat{p}_H(X) = \frac{1}{n} \sum_{i=1}^{n} K_H(X - x_i)$$

where H is a bandwidth matrix, symmetric and positive definite; K is the kernel function; and $K_H(X) = det(H)^{-1/2} K(H^{-1/2} X)$. The bandwidth matrix is in general a hyperparameter and it is possible to select it via cross-validation or similar strategies [Gre+05; MRL95]. After having obtained estimates for the involved densities, one can compute the MI in a similar fashion as the one described in the previous section. It is relevant to note that one may avoid computing a density estimate for the joint distribution by relying on Bayes' theorem or on the conditional entropy formulation for MI:

$$I(X;Y) = H(X) - H(X \mid Y) = H(Y) - H(Y \mid X)$$

### 2.3.3 Variational Approaches

A relatively recent trend in deep representation learning is employing variational-based objectives to train neural networks. This family of methods has been brought to deep learning relevance by Kingma and Welling [KW14] who contributed a general algorithm to train a hybrid neural network/directed graphical model architecture which they called a variational autoencoder (VAE). A similar model based on variational inference has also been developed by Rezende et al. [RMW14] under the name of Deep latent Gaussian model (DLGM). The attractiveness of these methods is their generative capabilities, as they can estimate[3] $p(X)$ for complex data spaces such as images. When compared with other modern generative models such as Generative Adversarial Networks (GANs) [Goo+14], their main benefit is the absence of a complex two-model game. Instead, the optimization objective is the evidence lower bound (ELBO) which can be reached[4] via stochastic gradient descent. This is a more stable objective when compared to the saddle point which is the global minimum for GAN models (an expository analysis of the GAN objective can be found e.g. in [Goo17]).

While the literature on this topic is vast and not especially related to fairness and invariance, it is still of relevance to the present discussion to introduce some of the main ideas behind variational inference. From there, I will touch upon a number of MI estimators which employ the same concepts while leveraging the well-known KL-divergence formulation of MI, also discussed here at the top of the current chapter.

Variational inference has been developed in the context of graphical mod-

[3]It is relevant to say that this estimation is implicit, meaning that a trained VAE can sample from $p(X)$, but point density estimates are not available.

[4]The usual caveats for non-conex optimization apply.

els as a way to perform approximate inference. The core idea behind the general methodology is to cast the inference process as an optimization problem. When given an intractable distribution $p$, the variational inference approach is to approximate the distribution with a class of tractable distributions $q_\theta$. The parameters will then be selected via search techniques, with most modern models employing stochastic gradient descent [RMW14; KW14]. A bare-bones, expository example for variational inference is to assume a simple directed graphical model $Z \to X$. If Z is representing the class or a feature of an example (e.g. "dog"), X is the observed data (e.g. images, possibly of dogs). Estimating $p(Z \mid X)$ can then be problematic. By Bayes' theorem, $p(Z \mid X) = \frac{p(X|Z)p(Z)}{p(X)}$; however, both the marginals involved are in general intractable in high-dimensionality settings. The variational inference approach here is to instead minimize some divergence measure between a known, tractable distribution $q(Z \mid X)$ and $p(Z \mid X)$. By taking the KL divergence, one can show the following:

$$\begin{aligned}
KL(q(Z \mid X) \mid\mid p(Z \mid X)) &= -\sum_z q(z \mid X) log \frac{p(z \mid X)}{q(z \mid X)} \\
&= -\sum_z q(z \mid X) log \frac{\frac{p(X,z)}{p(X)}}{q(z \mid X)} \\
&= -\sum_z q(z \mid X) log \frac{p(X,z)}{q(z \mid X)} \cdot \frac{1}{p(X)} \\
&= -\sum_z q(z \mid X) [log \frac{p(X,z)}{q(z \mid X)} - log\ p(X)] \\
&= -\sum_z q(z \mid X) log \frac{p(X,z)}{q(z \mid X)} + \sum_z q(z \mid X) log\ p(X) \\
&= -\sum_z q(z \mid X) log \frac{p(X,z)}{q(z \mid X)} + log\ p(X) \geq 0
\end{aligned}$$

Where the last inequality holds since the KL divergence is always positive. Taking the RHS we have:

$$
\begin{aligned}
log\, p(X) &\geq \sum_z q(z \mid X) log \frac{p(X, z)}{q(z \mid X)} \\
log\, p(X) &\geq \sum_z q(z \mid X) log \frac{p(X \mid z) p(z)}{q(z \mid X)} \\
log\, p(X) &\geq \sum_z q(z \mid X) log\, p(X \mid z) + \sum_z q(z \mid X) log \frac{p(z)}{q(z \mid X)} \\
log\, p(X) &\geq \mathbb{E}_{q(z|X)}\, log\, p(X \mid z) - KL(q(z \mid X) \mid\mid p(z))
\end{aligned}
$$

The RHS above is referred to as the Evidence Lower Bound or variational lower bound. Maximizing it is akin to maximizing the log likelihood of the data. Another way to interpret it is to see the following

$$
\begin{aligned}
KL(q(Z \mid X) \mid\mid p(Z \mid X)) = log\, p(X) \\
- (\mathbb{E}_{q(z|X)}\, log\, p(X \mid z) + KL(q(z \mid X) \mid\mid p(z))
\end{aligned}
$$

Thus, maximizing the ELBO implies minimizing the KL divergence between the intractable $p(Z \mid X)$ and the variational approximation $q(Z \mid X)$. When assuming a factorized $p(z)$, we are working in the space of *mean field* variational inference. VAE models parametrize the involved distributions $q(z \mid X)$ and $p(X \mid z)$ as an encoder and a decoder architecture respectively; $p(Z)$ can be any distribution in principle but it is easy to develop a closed-form loss function when employing Gaussian distributions (for details see for instance [Oda19]).

### 2.3.4 Mutual Information, Equitability and Noise

As discussed in this chapter, Mutual Information can be employed in different applications for the purpose of quantifying the strength of non-linear relationship between two random variables. In Information Theory, it is possible to build a connection between MI, channel capacity and the amount of noise in the transmission channel. This is relevant to the present discussion as it will build both an intuition and some theoretical foundation for the noise-based layer introduced later in Section 4.2.

A "transmission channel" in Information Theory is an abstract representation of a physical layer that can transmit information, or messages. Shannon [Sha48] showed how a noisy channel can still be employed for message transmission, even if the noise limits the *rate* at which messages can be sent and reconstructed, in the case of errors in the transmission, without errors. If one defines $X$ as the message sent and $Y$ as the message received, then it is possible to model the transmission channel by a probability distribution $p(Y \mid X)$. Shannon's noisy channel theorem defines the *capacity* of a channel as the supremum of the MI between $X$ and $Y$ over the possible distributions of $X$, which can be chosen by who sends the message:

$$C = sup_{p(X)} I(X;Y)$$

The result that bridges capacity, and therefore MI, to noise is the Shannon-Hartley theorem. Here one assumes that the channel is subject to additive Gaussian noise (measured in watts), and sees how the capacity is inversely proportional to it:

$$C = B \; log_2 \, (1 + \frac{S}{N})$$

Here, $B$ is the bandwidth of the channel in hertz and $S$ is the average signal power measured in watts. It is worthwhile to mention that Tishby et al. [TZ15] have employed a similar "noisy channel" model to study the learning dynamics of deep neural networks by noting that the same transmitter-receiver metaphor can be employed in neural network layers. More specifically, one can estabilish a neural layer $H_{i-1}$ as a transmitter, $H_i$ as the channel and $H_{i+1}$ as a receiver. Tishby et al. study how the measure $I(H_{i-1}, H_{i+1})$ varies over training time and only employ noise to obtain a well-defined mutual information, as in general the functional relationship between $H_{i-1}$ and $H_{i+1}$ would imply an infinite MI.

More recently, relating noise to mutual information has attracted interest when discussing the *equitability* property. Equitability has been first informally defined by Reshef et al. [Res+11]. In their paper, Reshef et al. discuss possible properties for measures that can relate different random variables together. In their informal discussion, equitability is the property of assigning similar scores to equally noisy relationships of different types. As an example, let us assume that $X$ and $Y$ are related as follows: $Y = f(X) + \eta$ where $\eta$ is a noise variate. Then, an equitable dependency measure $D(X;Y)$ should return the same value if we vary different invertible $f$ functions, such as linear or sine (when analyzed in the proper interval), while keeping the same $\eta$. Reshef et al. propose to measure equitability by measuring the $R^2$ value of different noisy functional relationships. The authors then propose an equitable statistic which is a normalization of mutual information, the Maximal Information Coefficient (MIC).

The contribution by Reshef et al. has attracted considerable attention, both positive and critical. Simon and Tibshirani [ST14] have noted how MIC displays low statistical power on some functional relationships; Kinney et al.

[KA14] expand on their definition of equitability. Critically, they propose two different definitions. $R^2$-equitability is achieved when the dependency measure is itself a function of the squared Pearson correlation:

$$D(X;Y) = g(R^2(f(X), Y))$$

Where $Y = f(X) + \eta$. The authors show $R^2$-equitability cannot be achieved by any non-trivial dependency measure; however, they also introduce the new notion of self-equitability, which requires that the dependency measure is symmetric and that

$$D(X;Y) = D(f(X);Y)$$

where $f$ is a deterministic, invertible function and $X \longleftrightarrow f(X) \longleftrightarrow Y$ is a Markov chain. It is worthwhile to notice that the definition does not specify a form for the noise model, contrary to the additive one defined above. Kinney et al. reason that this provides a more comprehensive setting for discussing the abstract concept of equitability. The authors also show that, while Mutual Information is self-equitable, the MIC of Reshef et al. is not.

The discussion regarding equitability is relevant when developing neural architectures that can reduce the MI between the learned representation and the sensitive attribute. As Kinney et al. highlight how an invertible deterministic $f$ applied to $X$ cannot change the MI w.r.t. $Y$, it is sensible to employ noise. This will be the topic of discussion in Section 4.2.

# Chapter 3

# Fairness in Classification and Ranking

Fairness in Machine Learning has attracted more attention as ML techniques become pervasive. In this context, AI researchers have been concerned with the ethics of the employed algorithms. One of the first contributions in the discussion has been developed by Friedman and Nissenbaum [FN96] in 1996. The authors reason that when decisions are made by "machines" without human intervention, there is a concrete risk of unfairness and discrimination. While this chance is unfortunately present also when dealing with human decision makers, a victim usually has the chance to appeal or discuss that decision. In their contribution, Friedman and Nissenbaum reason that "unfairness" should be defined with particular attention to systemic discrimination and moral reasoning. As an example, a random software error which impacts a single individual in a minority group is problematic on technical but not on moral grounds; on the other hand, a software which consistently displays negative bias towards minorities should be seen as unfair or discriminating. Since then, various authors (Verma and Rubin [VR18]; Mehrabi et al. [Meh+19] among the others) have coalesced around a definition which involves the concept of protected and unprotected groups. Simply put, a protected group is an observable group of people for which there is motivated concern of bias. Thus, a ML model is said to be "fair" if it does not discriminate against protected groups. This definition is purposefully generic: I will present specific definitions and how to evaluate them later in this chapter.

In the so-called "Big Data" era, where businesses both small and large are increasingly employing data analytics and Machine Learning [Nit19], there is new cause for concern. Namely, training data which displays biases will often cause models to reflect that bias, perpetrating it in the model. This is particularly troubling because it requires no actual discrimination from the individuals involved in training the models: simply by collecting data and training a ML model, there is the chance to reflect past biases in society. This is not an abstract concern, but a situation that has happened time and time again as ML and AI applications that leverage data attract more attention and scrutiny from both lawmakers and the general public.

One example is the aforementioned "beauty pageant algorithm" by beauty.ai [Lev16] which was trained to select the most "beautiful people" out of candidates which submitted pictures themselves. While the application itself is puzzling and problematic, its results were also biased, with a single dark skinned individual selected out of 44. When asked about the result, the researchers behind the algorithm pointed out the lack of training data for black and brown individuals. This outcome reflects the biases against women of color in human-judged competitions (see e.g. [Vla19]). Another example which attracted attention from the general public is the COMPAS software developed by Northpointe. This software evaluates the risk of "re-offending" for individuals who have been arrested and are awaiting trial. The software is made available to US judges and may be employed to inform the judge's decision about the individual being able to be released on bail (i.e. by paying a varying amount of money). ProPublica, a US-based consortium of investigative journalism, has performed a long-term analysis of the COMPAS risk scores [Ang+16] and has found evidence for bias against black people. The assumption here is that a high risk score (greater then 5 on the original COMPAS 1-10 scale - a binarization of sorts) would imply a "will re-offend" prediction; furthermore, the authors collected the criminal records of various individuals who have been evaluated by the software after 2 years and saw whether they did, in fact, commit another crime. The authors showed that the false positive rate for black individuals was much higher than the false positive rate for white people. Northpointe rebutted that they did calibrate their risk scores so that they were independent of ethnicity [DMB16], showing that systemic bias can still emerge in software even when designers and engineers take such issues into account.

In account of these issues, there has been a growing call, coming from both lawmakers and the general public, to have some guarantees of fair treatment in automatic decision making. Perhaps the most important regulation in the field has been put forward by the EU Parliament in the form of the General Data Protection Regulation (GDPR). This regulation concerns many concepts and procedures relating data processing for companies acting in the EU. Precise interpretation of the GDPR is beyond the scope of this thesis and the topic of active research. One recent contribution by Malgieri [Mal20] outlines how Recital 71 strongly specifies requirements for fairness in terms of non-discrimination. An excerpt follows:

> [...] In order to ensure fair and transparent processing in respect of the data subject, taking into account the specific circumstances

> and context in which the personal data are processed, the controller should use appropriate mathematical or statistical procedures for the profiling, implement technical and organisational measures appropriate to ensure, in particular, that factors which result in inaccuracies in personal data are corrected and the risk of errors is minimised, secure personal data in a manner that takes account of the potential risks involved for the interests and rights of the data subject, and prevent, inter alia, discriminatory effects on natural persons on the basis of racial or ethnic origin, political opinion, religion or beliefs, trade union membership, genetic or health status or sexual orientation, or processing that results in measures having such an effect.

Malgieri also focuses on the transparency requirements, citing again Recital 71:

> The data subject should have the right not to be subject to a decision, which may include a measure, evaluating personal aspects relating to him or her which is based solely on automated processing and which produces legal effects concerning him or her or similarly significantly affects him or her, such as automatic refusal of an online credit application or e-recruiting practices without any human intervention.
>
> [...]
>
> In any case, such processing should be subject to suitable safeguards, which should include specific information to the data subject and the right to obtain human intervention, to express his or her point of view, to obtain an explanation of the decision

> reached after such assessment and to challenge the decision.

The above recital is often summarized with the concept of a “right to an explanation” for individuals which are subject to automatic decision making and data processing. Therefore, at least two concepts seem to be required by the GDPR: non-discrimination and transparency. These two topics have been explored in the AI literature as “fairness” and “interpretability”.

The GDPR is not the only regulation put in place dealing with these matters. In France, the CNIL requires that “a fair algorithm should not end up generating, replicating or aggravating any form of discrimination”; in the US, credit scoring is regulated by the Equal Credit Opportunity Act:

> Statement of specific reasons. The statement of reasons for adverse action required by paragraph (a)(2)(i) of this section must be specific and indicate the principal reason(s) for the adverse action. Statements that the adverse action was based on the creditor’s internal standards or policies or that the applicant, joint applicant, or similar party failed to achieve a qualifying score on the creditor’s credit scoring system are insufficient.

In light of these concerns, a specific kind of model which has been increasingly adopted seems to be incompatible with the newly developed regulations. Deep neural networks are being employed in a plethora of applications and state of the art models often include millions of parameters [LBH15]. Convolutional object recognition models such as ResNets [He+16] sport as much as 101 layers and over 44 millions parameters. In this situation, isolating specific reasons for a model’s decision-making is very challenging. If making sense in a human-intelligible way of a model’s reasoning were possible, this would be the most straightforward path to vet a model: the unfair reasons and decisions would simply be discarded. Model explanation is indeed a very active area of research (see e.g. the survey by Adadi et al. [AB18] and the one by Carvalho et al. [CPC19]) and far from being solved. Another direction which is relevant to investigate is constraining deep neural network models so that they may be lawful, enabling them to be actually employed beyond research labs while limiting the possibility for discrimination.

Research on fairness in ML has developed multiple definitions of fairness. In this Chapter, I will review the ones relating to classification and ranking applications as they are relevant to understand and discuss the methodologies proposed in Chapters 4 and 5. Furthermore, I discuss the opportunities that representation learning algorithms provide in the context of fairness.

## 3.1 Definitions of Fairness

As previously discussed in this chapter and in the introduction, definitions of fairness in ML relate to the concepts of non-discrimination against historically disadvantaged groups. In this context, the relevant variables are $y$, the given ground truth; $\hat{y}$, the estimation given by a model $f$; and $s$, the sensitive attribute which holds values indicating whether an individual belongs to a historically discriminated group. Research on fair ML has focused on binary sensitive attributes, where $s = 1$ indicates belonging to a protected group. While most of the research has focused on classification scenarios, some work on ranking and regression has also been performed. As a starting point, I will borrow the three definitions of unfair treatment as summarized by Zafar et al. [Zaf+17].

**Disparate impact**. A decision system (automatic or otherwise) suffers from disparate impact when the decisions undertaken benefit (or hurt) a group of people more frequently than others. In real world scenarios, one such example is the Stop-and-Frisk program performed by the New York Police Department. The concept of such a program is to prevent crimes by stopping and interrogating people on the streets. Assuming that a civilian is benefited by avoiding police interrogation, black and latinx people have been damaged by this system overwhelmingly more than white people, per the data released by the NY chapter of the American Civil Liberties Union [ACL19].

**Disparate treatment**. A decision system suffers from disparate treatment when individuals which differ in their value of the sensitive attribute but are otherwise similar obtain different decisions. This is perhaps the most intuitive definition of "discrimination" in everyday language and is exemplified by the gender pay gap (see e.g. [WW05]).

**Disparate mistreatment**. A decision system suffers from disparate mistreatment when its error rates benefit (or hurt) a group of people more frequently than others. This is the situation in the COMPAS software, which mis-assigns high risk scores to black people more frequently than white people [Ang+16].

The concepts presented just may be transposed into formulas which may then be employed in classification and ranking. In the following, I will present a number of practical measures that are based on the above general definitions.

### 3.1.1 Fairness in Classification

Here I will present two measures of fairness which have been developed and employed by various authors in fair classification scenarios. I will also underline their connection with the definitions of unfair treatment as previously discussed in this chapter.

### Statistical Parity

Statistical parity in its simplest form requires that the following formula is true:

$$P(\hat{y} = 1 \mid s = 0) = P(\hat{y} = 1 \mid s = 1) \tag{3.1}$$

Where $\hat{y}$ is a variable representing the labels predicted by a classifier. Here, the assumption is that $y = 1$ are positive outcomes in a binary classification task (e.g. obtaining a loan). The formula may be generalized to allow a tolerance level $\epsilon$:

$$|P(\hat{y} = 1 \mid s = 0) - P(\hat{y} = 1 \mid s = 1)| \leq \epsilon \tag{3.2}$$

This metric has been employed by Dwork et al. [Dwo+12] and Zemel et al. [Zem+13] (who call it "discrimination") among the others. Statistical parity may be also employed to measure the difference in positive outcomes between individuals belonging to two different groups, simply by computing the difference in Equation 3.2. For this reason, it may be employed to evaluate disparate impact in a classifier.

### Equalized odds

Equalized odds evaluates the disparate mistreatment in a classifier. Plainly put, the classifier's false positive rate and false negative rate should be balanced among groups. In our running example of two groups, the two following equations need to be true:

$$P(\hat{y} = 1 \mid y = 0, s = 0) = P(\hat{y} = 1 \mid y = 0, s = 1) \tag{3.3}$$

$$P(\hat{y} = 0 \mid y = 1, s = 0) = P(\hat{y} = 0 \mid y = 1, s = 1) \tag{3.4}$$

As previously mentioned, the COMPAS software has been shown to be breaking this definition under a specific set of assumptions. This measure is employed e.g. by Zafar et al. [Zaf+17].

### 3.1.2 Fairness in Ranking

In ranking applications, the label $y$ often has a different meaning. It may either represent the position of an individual $x_i$ in a query or a relevance class $j$. Either way, the label has an ordinal meaning. Nonetheless, the concepts presented in Section 3.1 may still be employed. The field of fair ranking is relatively newer than fair classification and measures are still relatively scarce.

#### rND

The rND metric has been introduced by Ke and Stoyanovitch [YS17] in 2017. The metric is defined as follows:

$$\text{rND} = \frac{1}{Z} \sum_{i \in \{10,20,\ldots\}}^{N} \frac{1}{log_2 i} \frac{\mid S^+_{1\ldots i} \mid}{i} - \frac{\mid S^+ \mid}{N} \tag{3.5}$$

This metric computes the difference between the proportion of protected individuals in the top-i documents ($\frac{S^+_{1..i}}{i}$) and in the overall population ($\frac{S^+}{N}$). $Z$ is a normalization factor which is defined as the maximum possible value of the metric. A fractional log term is also included so to give more importance to imbalances in the top positions of the query. As argued by Ke and

Stoyanovitch, this metric can be seen as a generalization of the disparate impact/statistical parity concept in fair classification. It warrants mentioning that over-representation (higher rate than the population proportion) of protected individuals at the top of the list is also penalized by the metric.

#### Group-dependent Pairwise Accuracy

This metric is a generalization of the disparate mistreatment concept and has been introduced by Narasimhan et al. in 2020 [Nar+20]. Let $G_1, ..., G_K$ be a set of $K$ protected groups such that every document inside the dataset $D$ belongs to one of these groups. The *group-dependent pairwise accuracy* $A_{G_i>G_j}$ is then defined as the accuracy of a ranker on documents which are labeled more relevant belonging to group $G_i$ and documents labeled less relevant belonging to group $G_j$. Since a fair ranker should not discriminate against protected groups, the difference $|A_{G_i>G_j} - A_{G_j>G_i}|$ should be close to zero. In the following, we call the Group-dependent Pairwise Accuracy *GPA*.

## 3.2 Fairness in Representation Learning

When dealing with representation learning algorithms, the aforementioned definitions are all applicable. However, some authors in this literature have underlined an unique opportunity when it comes down to representation learning and fairness. McNamara et al. [MOW17] have drawn up a possible separation of concerns scenario which employs representation learning algorithms at its core. In their contribution, the authors explain how applications with fairness concern might employ a two-party scheme: user and regulator. The "user" here is any business which is interested into learning an automatic decision system that learns from individuals' data; the "regulator" is a second entity which employs the aforementioned data and debiases it, giving it back to the user. If the data truly contains no information about the sensitive attribute, the user may then employ any machine learning model without worry of unfair behaviour.

In this setting, the issue is of course how invariance might generalize to different fairness metrics such as statistical parity and equalized odds. However, there is still the issue of evaluating how much information is present in a random variable. While mutual information would provide a natural metric, as explained in Chapter 2, it is hard to estimate. In the literature (e.g. [Zem+13; Xie+17; Lou+15] among others) this is often evaluated as the difference between the accuracy of a classifier trained on the dataset $(\hat{x}, s)$ and the majority class performance on the same dataset. The rationale here is that a classifier finds no information about $s$ when it can do no better than predicting the majority class for each data vector $\hat{x}_i$.

When evaluating debiased representations in later Chapters, this is the approach that we will be employing.

## 3.3 Fairness Datasets

Evaluating fairness requires data for which there is a concrete concern that unfair classifiers might discriminate against certain groups. Therefore, one of the columns available has to be connected to an individual's protected characteristics such as gender and ethnicity. In this section, I will briefly present some datasets which have been employed in fair ML research.

### 3.3.1 COMPAS

The COMPAS dataset has been published by Propublica [Ang+16] as a long-term study of the same named tool by the US company Northpointe. This tool is made available to US judges who may use it to decide whether a person may be released on bail (i.e. by paying a fee) while waiting for trial. The reasoning here is that this opportunity should made available only to people at low risk of committing further crimes if released. The ground truth here is whether people have re-offended 2 years after their COMPAS evaluation, and the sensitive attribute is their ethnicity (white or black[1]).

### 3.3.2 Adult

The Adult dataset has been extracted from the 1994 US Census data by Kohavi [Koh96]. The ground truth here is represented by a simple indicator variable on whether an individual's yearly salary is over 50 000 US$. The sensitive attribute is an individual's gender.

[1]Employing the right words here is a complicated matter. I briefly explain my rationale here. While "ethnicity" usually refers to cultural aspects, what the COMPAS dataset has collected is a mixture of concepts (see [Ang+16]). In this dataset, people can fall into the groups "white", "black", "pacific-islander" and "other". Therefore, while this division is not about ethnicity per se, it definitely is not about skin color alone. Furthermore, I intend to avoid the term "race", which has little basis in scientific reasoning. Therefore, I chose the term "ethnicity" with some abuse of meaning. In the ProPublica article, "race" was employed. This is not meant as a critique to actors which employ this term with awareness: in the collective discourse, this word may be employed in a non-discriminating fashion, such as articles which may refer to "race relations". However, I prefer to try and move away from such terminology because of its loaded meaning and unscientificness.

### 3.3.3 German

The German dataset, short for German Credit Data, has been published on the UCI Repository [DG17] by Prof. Hofmann from the University of Hamburg. This dataset is a collection of credit-related variables such as employment and the presence of debtors; the task is to predict whether the individual is in a good credit standing (a binary variable). The sensitive attribute is an individual's gender.

### 3.3.4 Bank Marketing

The Bank marketing dataset has been published by Moro et al. [MCR14b] in the context of a data-driven study on the success of telemarketing campaigns. Here, the ground truth is whether an individual has subscribed to a term deposit, while the sensitive attribute is whether their age is above 40 years.

### 3.3.5 Law Students

The Law Students dataset contains information relating to 21,792 US-based, first-year law students and was collected to the end of understanding whether the Law Students Admission Test in the US is biased against ethnic minorities [WRC98]. This dataset has been first employed in fair ranking by Zehlike et al. [ZC19]. The task here is to rank law students based on their academic performance, and may be used to simulate a scenario where scholarships or some other kind of academic benefit needs to be assigned to a subset of individuals.

### 3.3.6 Wiki Talk

The Wiki Talk Page Comments dataset contains 127,820 Wikipedia comments which are labeled toxic or not toxic. A toxic comment can be defined as having "rude, disrespectful or unreasonable" content. This dataset has been employed in fairness when evaluating both classification [Dix+18] and ranking methods [Nar+20]. The term "gay" is commonly used as a sensitive attribute, since 55% of the comments labeled toxic contain the term "gay", while only 9% of the comments which do not have the term "gay" are labeled toxic [Dix+18]. The task of interest is therefore to provide a list of comments which are ranked from most to least toxic while taking into consideration the original, biased sorting.

# Chapter 4

# Invariance with Neural Networks

This chapter presents the idea of obtaining *invariant representations* w.r.t. a measurable disturbance factor in a neural network. Methods both original and already present in the literature will be displayed. Invariant representations, as mentioned in the Introduction, have the following benefits:

- *Transferability across domains.* In the field of multi-task learning, it is a well known fact that some tasks (datasets) may be related and that neural representations extracted from one can also be useful in another (see e.g. Caruana [Car97]). However, removing task-specific information can be beneficial in Domain Adaptation, where the assumption is that the labels for future tasks are unavailable.

- *Fairness.* As described in the previous Chapter, the issue of fairness in Machine Learning may be addressed as discarding sensitive data and *actively removing* their correlates in the feature vectors. Neural representations that are invariant to sensitive attributes enable any downstream learning algorithm to be fair "by construction".

The chapter is organized as follows. First, I will briefly review the core ideas that led to the interest in neural representations and invariance, also reviewing the relevant approaches in previous literature 4.1. I will then present my original work on a general noise layer that benefits invariance (Section 4.2) and its applications on fair classification (Section 5.1), fair ranking (Section 5.2), fair and *interpretable* learning (Section 5.3). Another methodology for shared/invariant representations is then described in Section 5.4.

## 4.1 Neural Networks and Neural Representations

In this chapter, I will refer to networks as functions $f_\theta(x)$ where $x$ is a vector of features and $\theta$ are real-valued parameters. The topics presented in this chapter fall under the supervised learning paradigm, where data is available in the form of $n$ example-label pairs $(x_i, y_i)_{i=1}^n$. When relevant, we will extend the available data to the triplets $(x_i, y_i, s_i)_{i=1}^n$ where $s$ is a measurement of a disturbance factor or sensitive information. When relevant, the output of the $i$-th layer will be written as $f_\theta^i$, with a slight abuse of notation w.r.t. the parameter set. The task here is one of obtaining, as previously mentioned, an invariant *representation* w.r.t. $s$ which still has as much information as possible about $x$ and $y$. Before discussing more deeply this objective, it is worthwhile to give context about how the concept of neural representations has developed in the literature.

### 4.1.1 From the Artificial Neuron to Distributed Representations

The idea of constructing artificial neural networks (NNs) has a long history, dating back to the development of the artificial neuron in 1943 by McCullogh and Pitts [MP90]. In this representation, a neuron's dendrites are represented by a number of different input boolean values. A neuron can either be activated, outputting a value of 1, or not, outputting a value of 0. This kind of neuron can only be employed to represent logical gates and functions, and the first ante litteram[1] machine learning application of neural networks is due to Frank Rosenblatt's development of the perceptron in 1958 [Ros58]. In modern terms, we would call the perceptron a supervised learning algorithm which learns a linear, binary classifier. The input vectors $\mathbf{x}$ may be real valued, as the weights for the input $\mathbf{w}$ and the bias value $b$. The output value $f(\mathbf{x})$ for the neuron is a simple threshold:

$$f(x) = \begin{cases} 1 \ \mathit{if}\, \mathbf{wx} + b \geq 0; \\ 0 \ \mathit{otherwise} \end{cases} \tag{4.1}$$

The learning algorithm is iterative and modifies the weights at each step depending on each $w$'s contribution to the error. The perceptron is unable to converge when the data $(\mathbf{x}, \mathbf{y})$ is not linearly separable, a point that Minsky and Papert ("Perceptrons", 1969 [MP69]) stressed by pointing out the algorithm's inability[2] in learning the rather simple exclusive-or logical function. Circumventing this issue would require multiple breakthroughs. The first in-

[1]The definition of "Machine Learning" is often attributed to Arthur Samuel [Sam59] who published the description of a self-training algorithm for the game of checkers in 1959, the year after Rosenblatt's paper about the perceptron algorithm.

[2]One aspect that is often left unsaid about the matter is that a perceptron can learn

tuitive insight is that multiple perceptrons can be connected "horizontally" so to build two layers of multiple input and output neurons; furthermore, it is possible to add another "layer" of neurons at the end of this structure, obtaining what is often called, with some degree of inaccuracy[3], a multi-layer perceptron (MLP). In this kind of architecture, we have three layers of artificial neurons, which are often called input, hidden, and output layer, and two set of weights $w_1$ (connecting the input to the hidden layer) and $w_2$ (connecting the hidden layer to the output layer). Employing Rosenblatt's training algorithm would require computing the error contributions for the first set of weights $w_1$, which is not straightforward. The necessary breakthrough here is the backpropagation algorithm, which makes it possible to propagate error gradients from the output layer to the preceding ones. The invention of backpropagation is a contentious matter. Rumelhart, Hinton and Williams [RHW86] are often credited for the discovery. While it is certain that the authors managed to establish its usefulness in *learning representations* in the hidden layers of a multi-layer perceptron in 1986, the idea of employing an efficient algorithm for automatic reverse-mode differentiation over multiple applications of different functions is due to Linnainmaa in 1970[4] [Lin76]. The

the XOR function if an extra feature is given which is 1 iff the two other features are 1. This is indeed included in Minsky and Papert's work and and also cited by Rumelhart, Hinton and Williams in 1986 [RHW86].

[3]As a perceptron has a single output neuron, building multiple layers of perceptrons would employ the output neuron as the input for other neurons; instead, what is called a multi-layer perceptron is a stack of perceptrons in both a horizontal (more perceptrons, side by side, building a output layer with multiple neurons) and vertical (the previously mentioned output layer can be connected via other weights to another layer, which in turn becomes the output layer) direction.

[4]The first English article was indeed published by Linnainmaa in 1976 [Lin76] while his master thesis which first introduced the aforementioned algorithm was written in Finnish

first application of efficient reverse-model automatic differentiation to a neural network-like model is due to Werbos in 1982 [Wer82]. Nonetheless, the interest in neural architectures was renewed by the possibility of employing multiple layer of neurons. Furthermore, Cybenko [Cyb89] showed in 1989 that calculating each neuron's output by means of a sigmoid function makes it possible to approximate any function via a MLP architecture. The result would be then extended to other (non-polynomial) activation functions by Leshno et al. [Les+93] in 1993. Putting this issue aside, Rumelhart Hinton and Williams do contribute the central idea of *internal representations* which may emerge from the training of a neural network in the form of the activation of a hidden layer's output. This concept, also explored in depth by Hinton in "Distributed Representations", 1984 [Hin], may be informally described as a process of automatic feature extraction, where an algorithm learns independently combinations of the original features which are useful to the task at hand. This effect is particularly apparent in networks employing convolutional parameter sharing (also called convolutional neural networks or CNNs) and trained over image data. In these networks the neural activations can be thought of as being organized in a 2-dimensional grid. The work of Yann LeCun et al. in 1990 [LeC+90] showed that a CNN trained via backpropagation could recognize handwritten digits with a high degree of accuracy while building internal representations that may be easily visualized and represent various features of the original input. It warrants mentioning that the backpropagation algorithm has no understanding of what is interpretable to the human eye. The visualizable features are therefore an emergent property of backpropagation-trained networks. In the following chapters, I will refer as a "neural representation", or simply representation, as the output of a neural

in 1970 [Lin70].

network architecture at some layer $i$. We will now move to a brief review of some relevant neural architectures and their training algorithms.

### 4.1.2 Modern Feedforward Networks

Modern architectures which are based on the multi-layer perceptron are being widely used for a plethora of applications, from computer vision to natural language processing. The base building block for this kind of applications is the fully-connected layer, in which all neurons at layer $i-1$ are connected to all neurons at layer $i$. Such a model may be trained with stochastic gradient descent, in which the error function gradient is approximated. This may be done by first propagating forward the activations and predictions for a subset of training examples (usually called batch or mini-batch); error gradients can then be computed and back-propagated through the architecture. The error gradient computation depends on the definition of a *loss function*, i.e. a training objective. For network which are employed in classification, the cross entropy loss is usually employed:

$$J = -\sum_{x \in X} p(x)\ log\ q(x)$$

Here $p(x)$ is the given class label distribution for example $x$, and $q(x)$ is the estimate given by the neural network. A network can output a probability distribution by employing a softmax function in its last layer:

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

Where $\mathbf{z}$ is the vector of non-activated neuron outputs at the last layer. This guarantees that the activated values sum up to one and the vector of activations may be interpreted as a probability distribution over the class labels. Similar models have also been used in fair classification.

When one is performing ranking, the given labels $y$ have an ordinal meaning. They appear in the form $y \in \{0, 1, ..., K\}$ where $K$ can either be the number of relevance classes or the position of a document in a result query (in this case, $K = N$, the number of documents in the query). The loss function and overall architecture depends on the ranking strategy employed. A pointwise ranker assignes a probability to each document independently; a pairwise ranker compares two documents $x_1$ and $x_2$ and outputs the $(y_1 > y_2)$ probability; a listwise ranker takes full queries into its input and computes a loss function which depends on the overall ordering. As an example, the listwise ranker developed by Cao et al. ([Cao+07b]) compute the top-1 probability for each document and then employ cross entropy.

### 4.1.3 Neural Invariance

Invariance in neural representations has emerged as a possible strategy to construct fair-by-construction machine learning algorithms. While other fairness methodologies focus on constraints on *predictions*, this family of methods is centered on constraining *representations*. In this setting, the concern is not about directly modeling $p(y \mid s)$ but to reduce the information content of $s$ in a predefined layer of a neural network, $f_\theta^i$. As discussed in Chapter 2, a general definition of "information" is complex. While reducing the Mutual Information between the sensitive attribute $s$ and the obtained representation $\hat{x}$ seems to be a sensible target, one has to rely on estimation or variational approximations. One could then target distributional distances by reasoning that different values of $s$ imply different distributions, and matching $p(x \mid s = i)$ with $p(x \mid s = j)$ would also lead to invariance. Aside from simple moment-matching, more complex distributional distances are parametrized by an hypothesis class.

There are a number of existing methods in the literature which are relevant to the topic at hand. In this section, I introduce the ones which are relevant to better contextualise my contributions.

### 4.1.4 Gradient Reversal for Domain Adaptation

Ganin's Gradient Reversal Layer ( [Gan+16]) has emerged as a tool in unsupervised domain adaptation. Domain adaptation is a task belonging to the "transfer learning" field where the objective is to train a neural network architecture so that it employs data from a source task $(x_i, y_i)_{i=1}^{n}$ and a target task $(x_j, y_j)_{j=1}^{m}$ so that the performance is maximized on the target task. Usually, $n >> m$: the rationale for the task is to employ data which is abundantly available (the source task) to compound scarcely available information (the target task). More in general, transfer learning with neural networks has been investigated by various authors in the last decade as a way to investigate the transferability of learned representations. One of the more seminal papers in this area is "CNN Features Off the Shelf: An Astounding Baseline for Recognition" by Razavian et al. [Sha+14]. The authors employed an AlexNet model [KSH12] pretrained on ImageNet by extracting its neural activations after the last convolutional layer. Employing non-neural classifiers trained with these activations as training vectors shows impressive performance on unrelated datasets. This fact seems to suggest that neural networks trained via backpropagation are able to extract features which display strong generalization properties, and has kickstarted the topic of deep transfer learning (see e.g. the 2018 review by Chuanqi et al. [Tan+18]).

In this context, the contribution by Ganin et al. is to perform transfer learning in a particular setting, where one has a source and target task which are known to be related. The running example the authors provide is the one of online product reviews with different categories. Assume that an online shop which traditionally focused on books has trained a classifier to perform sentiment analysis on text-based reviews; the same shop now wants to provide the same service for a newly introduced category of products, such as consumer electronics. Reviews are just now flooding in, and labeled data is still scarce. This is the more specific setup where domain adaptation techniques may be applied successfully.

In this context, Ganin et al. contribute a domain-adversarial classifier. The base idea is to employ a pair of sub-models $f_y$ and $f_d$ connected to a number of shared layers $f$. I will keep the original notation by Ganin et al. in which domains are noted as different values of a domain variable $d$; furthermore, I will refer to the connection weights for these models as $\theta_y$, $\theta_d$ and $\theta$ respectively. Each sub-model is optimized via gradient descent, maximizing $p(\hat{y} \mid x)$ and $p(\hat{d} \mid x)$ respectively; however, the last layer in $f$ is connected to the first layer of $f_s$ via a "gradient reversal layer", i.e. a layer which computes the identity function during forward propagation and the $f(x) = -x$ function during backpropagation. The rationale for this layer is to find an equilibrium/saddle point between two differing objectives: classifying accurately on $y$ and reducing domain-specific information. The sub-network $f_d$ is therefore employed as a "domain regularizer" and its gradient information, when reversed, may be used to reduce the divergence between the features extracted from training vectors belonging to the target and the source domain. Ganin et al. outline how this procedure is related to different distributional distance measures. One possible formulation of this problem is to find the parameters which deliver a saddle point for the following min-max problem:

$$\begin{aligned}\hat{\theta}_y, \hat{\theta} &= \arg\min \ L(f_y) \\ \hat{\theta}_d, \hat{\theta} &= \arg\max \ L(f_d)\end{aligned}$$

Methods based on Ganin et al.'s gradient reversal layer have also been employed in fairness, e.g. by Xie et al. [Xie+17]. It warrants mentioning that this methodology is however focusing on performance on the target task. While the concept of invariance is of course useful as it minimizes domain divergence, it is not the objective of the contribution. Recently, Zhao et al. [Zha+19] have highlighted how invariant representations are not a sufficient condition for domain adaptation, contributing an example where invariance may instead yield a higher error on the target task. Therefore, there is some divergence between objectives when dealing with domain transfer and invariant representation learning.

Having introduced the relevant architectures and methodologies, I now come to my contributions. Inspired by Ganin et al.'s work, I have developed a new differentiable layer which employs noise to further reduce the mutual information between a learned representation $\hat{x}$ and a sensitive attribute $s$.

## 4.2 A Noise Module

In this section I will present how to develop a differentiable "noise layer" which is useful in learning invariant representations. This contribution has been published in the ACM Symposium for Applied Computing 2020 [CEL20] in collaboration with Dr. Laura Li Puma and Prof. Roberto Esposito.

Encouraging invariance w.r.t. a nuisance factor or a sensitive attribute $s$ is, as discussed extensively in Chapter 2 and in the previous section, a complex matter. While removing the aforementioned attribute from the dataset is a sensible starting point, it does not constitute a solid solution. The issue is that it is still possible to glean information about $s$ from the remaining features. One example is provided by the existence of the gender pay gap (see e.g. [WW05]), which makes it evident that there may be correlations between salary and gender in datasets relating to loan acceptance.

In neural networks, features are neural activations constructed by way of complex non-linear combinations of the given input vectors. Thus, removing information from the internal representation of a network warrants ad-hoc learning mechanisms, some of which have been introduced in the previous section. As domain adaptation algorithms do not need to remove all domain information (and this objective in general may not be desirable, see e.g. Zhao et al. [Zha+19]), there might be the need to also adapt neural architectures to the task of information removal.

To provide a motivating, if hyperbolic, example, assume some feature $x_i$ has a high degree of correlation (informally) w.r.t. $s$, so that information about $x_i$ has to be completely removed to obtain an invariant representation. Furthermore, $x_i$ is a good estimator of $y$: it follows that learning to estimate $p(y \mid x)$ via backpropagation would lead to representations that still contain information about $x_i$ and, by proxy, about $s$. Provided that the training algorithm does encourage removing information about $s$ via an explicit training penalty - as in Ganin et al.'s domain regularizer [Gan+16] - or otherwise - as in Louizos et al.'s fair disentanglement [Lou+15] - there exist at least two different ways to set neural weights so that information about $x_i$ is removed.

- *Forcing non-injectivity.* That is, the network is not sensitive to changes in $x_i$. With a slight abuse of notation, let us define a two-argument network $f(x, x_i)$. Non-injectivity here would imply that $f(x, a) = f(x, b)\ \forall (a, b) \in x_i$.

- *Reducing Mutual Information.* In Chapter 2 and earlier in this chapter, I discussed how a possible definition for an invariant representation $\hat{x}$ is null mutual information w.r.t. $s$: $I(\hat{x}, s) = 0$. This would imply that $p(\hat{x}, s) = p(\hat{x})p(s)$, i.e. the two random variables would be independent.

While it stands to reason that domain adaptation and fairness training objectives might lead to non-injectivity and null mutual information, it is not obvious how a neural network model might encourage these objectives in its architecture, i.e. the connection patterns between layers. As an example of how connection patterns can help in pursuing specific learning objectives or domains, let us consider convolutional neural networks. Convolutional neural networks, employ convolutional layers which extract features in a locally-sensitive way. This idea is suitable for image data, in which pixel locations are meaningful and give rise to higher-order concepts such as shape, contour and texture. Developing a similar concept for the idea of invariance would give a network a simpler way to remove problematic features from the computation of internal representations. As further motivation, it is relevant to investigate more deeply into how a fully-connected network could learn to be "non-injective" or to reduce mutual information. Let us look at the first and second layers $f^1$ and $f^2$ of a neural network, with no assumptions on depth. For now, let us also abstract away the loss function and training objective details, assuming that a suitable choice has been made. As for the activation function, let us assume for the sake of simplicity in the discussion that the identity function has been employed. Therefore, all neurons at the second layer activate as a linear combination of the input features:

$$o_j = \sum_{i=1}^{k} w_{i,j}\, x_i$$

Now, if we wish the activation vector $\mathbf{o} = f^2(\mathbf{x})$ to not contain information about the problematic feature $x_i$, the simplest solution is to set all the weights $w_{i,j}$ to 0. Other solutions might be possible, and could produce representations that are "invariant enough" for a given task such as domain adaptation; however, in this extreme case where it is not possible to employ $x_i$ and still be invariant w.r.t. $s$, the aforementioned solution is the clearest way to discard $x_i$.

Coming to the objective of reducing mutual information, the issue here is two-fold. Employing a deterministic, invertible function to transform a random variable $X$ does not impact its MI w.r.t. a nuisance factor; while neural networks are in general not invertible, it would be beneficial to instead design a stochastic procedure. Secondly, while Tishby et al. [TZ15] have contributed an analysis of MI between neural network layers and have shown how the training of neural networks via stochastic gradient descent displays a compression phase where information about the input random variable $X$ is discarded, more recent work by Saxe et al. [Sax+19] has shed light on the fact that a plethora of factors may impact such an analysis. In light of the fact that MI is in general unaffected by deterministic invertible transformations, Saxe et al. show how the aforementioned compression phase is not displayed by networks which employ non-saturating activation functions. Furthermore, another contribution by Goldfeld et al. [Gol+19] has shown how the compression phase of Tishby et al. might also be explained by a "clustering effect" of representations instead of actual changes in the MI between layers. Simply put, neural representations of vectors belonging to the same class are clustered together throughout the training process.

Having analyzed two limitations of current neural architectures when employed in the learning of invariant representation, I now come to the proposal of a noise layer [CEL20] with a one-to-one connection pattern. That is, the number of input and output neurons for this layer are not a hyperparameter but are fixed to be the same. For ease of notation, I will assume that a noise layer is inserted as the very first layer in the architecture, therefore having features from $\mathbf{x}$ as its output and particularly the feature we are assuming to be problematic, $x_i$. For each of its output neurons, this layer computes the following function:

$$o_i = g(w_{i,1}x_i + w_{i,2}\eta)$$

The number of output neurons for the layer is the same as the number of input neurons, displaying one-to-one connections. This layer has $i \cdot 2$ parameters, with one set simply multiplying the previous layer's neural activation for the $i$-th neuron, while the second one regulates the amount of noise $\eta$ which gets summed to the previous computation. Here any kind of noise might be used; my experimentation has focused on uniform ("white") noise. A differentiable non-linearity $g$, which may be chosen by the user, is also present in the formulation. A schematic of this layer, comparing it to a fully connected layer, can be seen in Figure 4.2.

The comparison with a fully connected layer is relevant as it shows how the layer's formulation is tackling the two issues with existing neural architectures when employing them to learn invariant representations. Firstly, eliminating a problematic non-sensitive feature $x_i$ and achieving non-injectivity

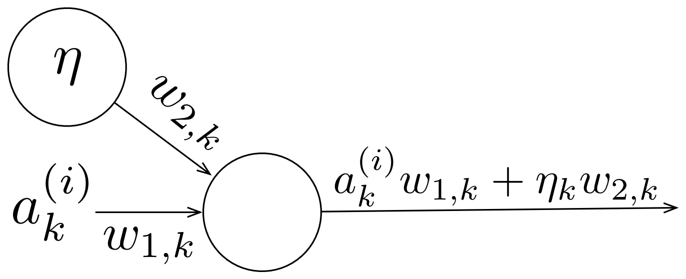


Figure 4.1: Schematic of the $k$-th component of a noise layer inserted at depth $i$ in a neural network architecture. Here, $a_k^i$ is the activation of the $k$-th neuron of layer $i$. Each neural activation is multiplied with a weight and an adjustable amount of noise is added. A non-linearity (not shown in the figure) is also added.

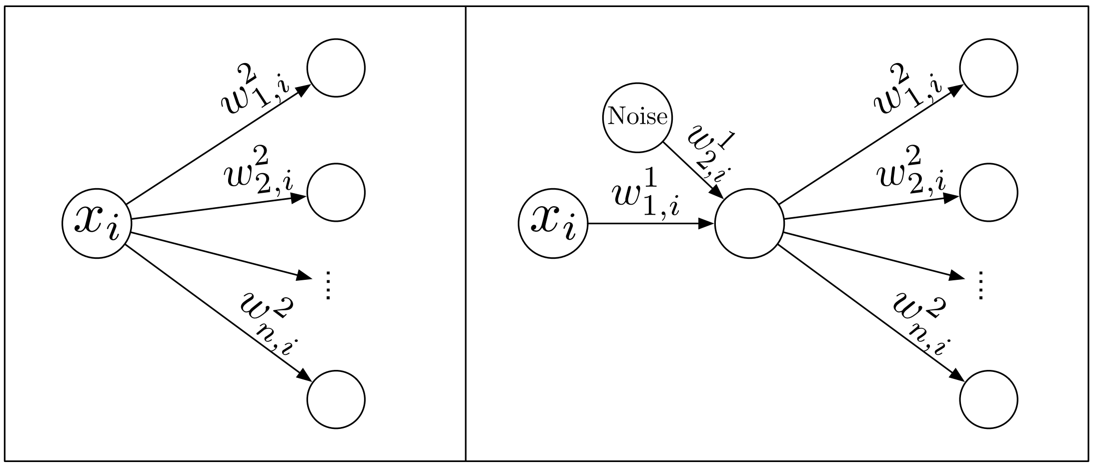


Figure 4.2: The noise layer (right) proposed by Prof. Roberto Esposito, Dr. Laura Li Puma and I [CEL20], compared with a regular fully connected layer (left). The one-to-one connection pattern allows for easier suppression of problematic features, and adding an adjustable amount of noise helps with controlling mutual information over the representations.

requires setting to 0 all the weights connecting it to the neurons in the second layer. Employing a one-to-one connection scheme allows for feature-wise dampening, allowing a single weight $w_{i,1}$ to be set to 0. It is relevant now to generalize this discussion to activation functions beyond identity, as employing differentiable non-linearities is necessary to have universal approximation, as discussed previously in this chapter and as discovered by Cybenko et al. [Cyb89]. Taking into consideration the sigmoid function:

$$\theta(x) = \frac{e^x}{e^{x+1}}$$

its value is 0 only in the limit where the input goes to $-\infty$. Thus, a problematic feature $x_i$ can never really be removed; however, it can be propagated as an arbitrarily small value as the relevant weights become more and more negative and the function gets "saturated" negatively. Non-injectivity may also be achieved if the sigmoid is saturated positively. Here, the propagated value would always be 1, clustering together the activations for different values of $x_i$. A similar result is valid for the tangent hyperbolic (tanh) activation function:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

which is instead defined between -1 and 1. Here, positive saturation, negative saturation and null output are all useful results from a non-injectivity perspective. However, obtaining an output which is always 0 seems to be challenging as the function gradient is highest around 0. Other activation functions such as the rectified linear unit (ReLU) appear to be more challenging to employ in the invariance setting. Taking into account the aforementioned results by Saxe et al. [Sax+19], which show how the MI between the layers of a network is not decreasing when employing a non-saturating function such as the ReLU, we have focused our experimentation to sigmoid functions as the non-linearity of choice for the noise layer.

Coming to the idea of employing a learnable amount of noise, this concept stems from the contributions in statistical theory dealing with the concept of "equitability" and mutual information (see Chapter 2.3.4). Kinney et al. [KA14] have highlighted how MI is invariant to deterministic transformation as it is self-equitable. Therefore, our noise module allows searching the space of stochastic transformations where the mutual information between the learned representations $\hat{\mathbf{x}}$ and the nuisance variable $s$ is reduced.

The noise module presented insofar is fully differentiable and might be inserted in any neural network model. However, some discussion is warranted about employing noise. Convergence via gradient descent is in general not guaranteed when employing randomly sampled variables, as any employed loss function $\mathcal{L}$ would then not be functional (univalent) - i.e., it would associate different outputs to the same input value, depending on the sampled noise. We circumvent this problem by sampling only once, at the beginning of the learning process. The values of $\eta$ are uncorrelated with $s$, but we do not break the functional loss assumption and can therefore expect gradient descent to converge. The insertion of noise during neural network training has previously attracted attention in the context of generalization. Srivastava et al. have contributed the Dropout methodology [Sri+14] as a way to prevent overfitting in over-parametrized neural networks. Dropout is indeed a "noise layer" as it samples from a Bernoulli distribution during training and multiplies each neuron's activation with the trial's result. Informally, this can be thought of as randomly "shutting down" neurons in a specific layer during training. In their experiments, Srivastava et al. show how this helps in reducing neuron co-adaptation and helps in preventing overfitting behavior. Li et al. [LXL16] contributed a generalization of the Dropout technique (called Whiteout) which instead is sampling from a Gaussian distribution, adding its value to neural activations. Here, the authors show that even when employing noise the loss function can be proven to converge *in expectation* to the true one. Our strategy is instead to sample once, keeping the loss function consistent. Furthermore, this allows for the training and inference procedures to be the same, whereas Dropout and Li et a.'s Whiteout needs to be "turned off" at test time.

As a summary, the noise layer provides two separate mechanisms which are tailored on the problem of information removal.

- *Feature-wise dampening.* Each feature or neural activation is weighted by a *single* weight $x_i$. This provides a more efficient mechanism in removing or dampening problematic features when compared with fully connected layers.

- *Feature-wise noise masking.* Each feature is mixed with an adjustable amount of noise $w_2 \cdot \eta$. Thus, it is possible to avoid the theoretical limits of reducing MI via deterministic transformations due to equitability (see Section 2.3.4).

To provide a first investigation of the noise module's capabilities in learning invariant representations, we tested it on a toy dataset in which there actually is a highly problematic feature which displays a functional relation with the sensitive attribute. This dataset is defined as follows:

$$\begin{aligned} s &\sim \mathcal{B}(0.5) \\ x_1 &\sim \mathcal{U}(0, 50\,000) \\ x_2 &= 20\,000 \cdot s \\ y &= \mathbb{I}_{x_1+x_2>35\,000} \end{aligned} \tag{4.2}$$

This dataset represents a toy "loan assignment" problem where $x_1$ and $x_2$ are two salary-related variables: the ground truth $y$ simply looks at whether their sum is over a threshold. Here, it is relevant to note that $x_2$ is the problematic feature. This is because it is a simple function of $s$, which is sampled from a Bernoulli distribution. $x_1$ is sampled uniformly and may be employed to classify $y$ fairly. This looks like an easy enough task for any learner, which has only to ignore $x_2$ to classify accurately (somewhat) and fairly. Of course, in a real world scenario simple feature selection and preprocessing would avoid the issue altogether. Furthermore, while this dataset may represent a very hyperbolic "gender pay gap" situation, the issue is much more nuanced in real world data. However, the argument here is that a fair invariant learner must be able to handle such a clear cut situation. We experiment with networks that include a noise layer as their first input, varying the size of a second layer from 4 to 16. The architecture and training objectives are set up as in Ganin et al.'s domain adversarial network [Gan+16]. What we see is that networks employing a noise layer are able to learn a representation $\hat{x}$ which contains little information about the sensitive attribute. This can be observed in Figure 4.3, where we show classifier accuracy for a random forest model predicting $s$ when trained on the learned representations . The vertical axis displays the MI estimated by a state of the art methodology by Belghazi et al. [Bel+18]. It is apparent that employing a noise layer

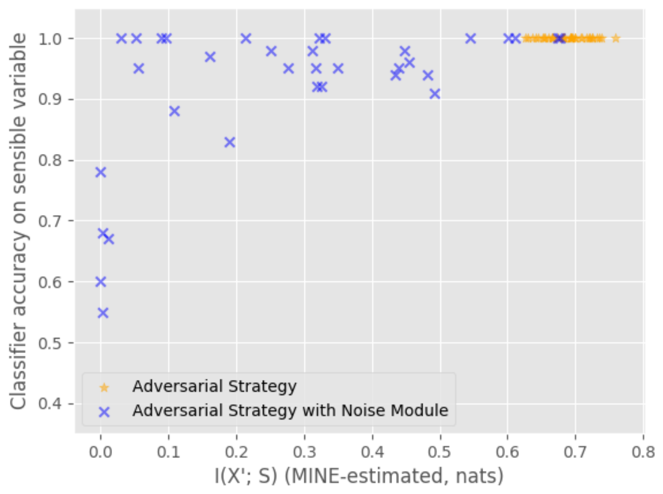


Figure 4.3: Performance of networks trained on the dataset defined in Equation 4.2. The horizontal axis represents the mutual information between the learned representations $\hat{x}$ and $s$ estimated with the method by Belghazi et al. [Bel+18]; the vertical axis represents classifier accuracy when trained on the pairs $(\hat{x}, s)$. The relationship between the two is noisy, but positively correlated; furthermore, architectures containing the noise layer were able to decorrelate w.r.t. $x_2$, the problematic feature.

reduces the estimated MI and that low levels of MI correlate to invariant representations w.r.t. $s$.

Starting from these results, we applied the noise module to different tasks in the fairness framework. At first, we tackled the fair classification problem, showing superior performance on invariant representation learning; secondly, we also employed this mechanism to investigate two separate fair ranking mechanisms. These contributions are analyzed in the next chapter.

# Chapter 5

# Fairness with Neural Networks

In this chapter I will discuss how the methodologies introduced in this thesis may be employed to learn invariant representations and fair neural network models. My contributions have touched upon fair classification and fair ranking with different models and methodologies. I will start from applying the noise module to the fair classification problem, then moving on to a contribution in pairwise fair ranking. The last two sections of this chapter touch upon a framework for interpretable fair learning and an invertible neural network model.

## 5.1 Fair Classification with a Noise Module

Fair classification is the task of learning a classifier $f$ while constraining the results for fairness. As discussed in Chapter 3, there are a number of fairness definitions. Learning invariant representations enables a multi-agent scenarios where representations are debiased by a third party before being given back to the end user. Thus, if no information is present in the representation about the sensitive attribute, even malicious actors would be unable to actively discriminate against certain groups. In this Section, I present how a noise module can enable a feedforward neural network trained by back-propagation in learning representations that are truly invariant w.r.t. to a sensitive attribute.

The starting point for this work is the gradient reversal concept as developed by Ganin et al. [Gan+16]. Thus, we employ three networks. One main network $f_\theta$ which will learn the invariant representations $\hat{x}$; two sub-networks $f^y_{\theta_y}$ and $f^s_{\theta_s}$ which are connected to the last layer of $f_\theta$. The gradients for $f^s_{\theta_s}$ are reversed when back-propagated through the rest of the network. The overall training objective is therefore as follows:

$$\hat{\theta}, \hat{\theta}_y, \hat{\theta}_s = \arg\min_{\theta,\theta_y}[L(y, f^y(f(x)) + \lambda \max_{\theta_s} L(s, f^s(f(x)))] \tag{5.1}$$

The general structure of the methodology and the gradients employed to optimize each network and sub-network can be found in Figure 5.1.

We compare networks trained with regular fully connected layers with other networks which include a noise module as their first layer. A representation of the latter type of networks can be seen in Figure 5.2.

### 5.1.1 Experiments and Results

We experimented on the Adult, COMPAS, Bank and German datasets (see Section 3.3, adopting a 60-20-20 train/validation/test split. The experimental procedure is as follows:

- Each model is trained on the original vectors $\mathbf{x}$ from the dataset.
- The invariant representations $\hat{\mathbf{x}} = f_\theta(\mathbf{x})$ are extracted from the model.

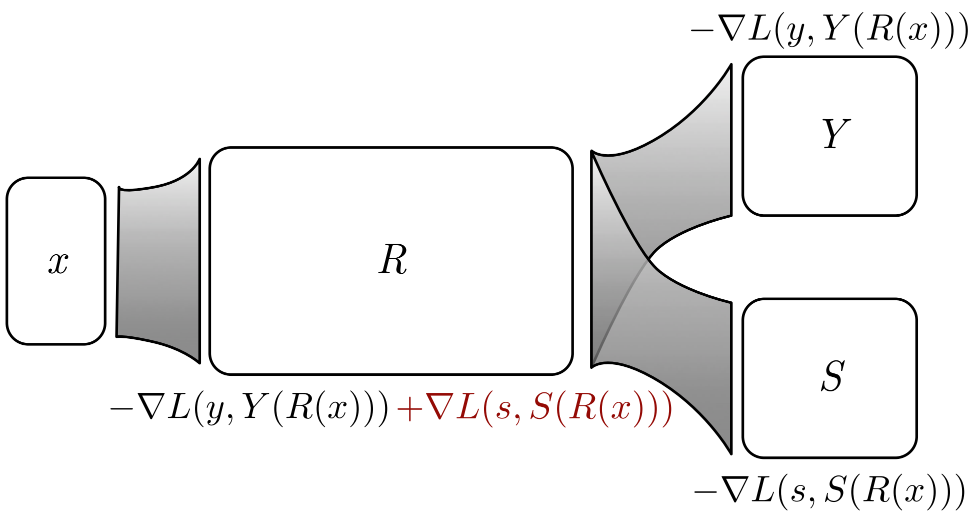


Figure 5.1: A schematic representing the main network and the sub-networks employed to optimize for invariant features. Adjacent to each component the employed gradients are reported. Note that the direction of the gradient for $f^{y}_{\theta_y}$ is inverted when applied to the main network $f_\theta$.

- Logistic regression and random forest algorithms are trained on $(\hat{\mathbf{x}}, y)$ and $(\hat{\mathbf{x}}, y)$.

We optimized the following hyperparameters:

- *Fairness importance*, i.e. the accuracy/fairness tradeoff parameter $\lambda$ in Equation 5.1.
- *Learning rate*, varying from $10^{-}5$ to $10^{-}2$.
- *Number of hidden layers*, varying from 1 to 3.
- *Number of epochs*, starting from 100 and up to 3000.

We report experimental results for the models that include a noise layer as their first and regular adversarial models in Figures 5.3 through 5.6. The

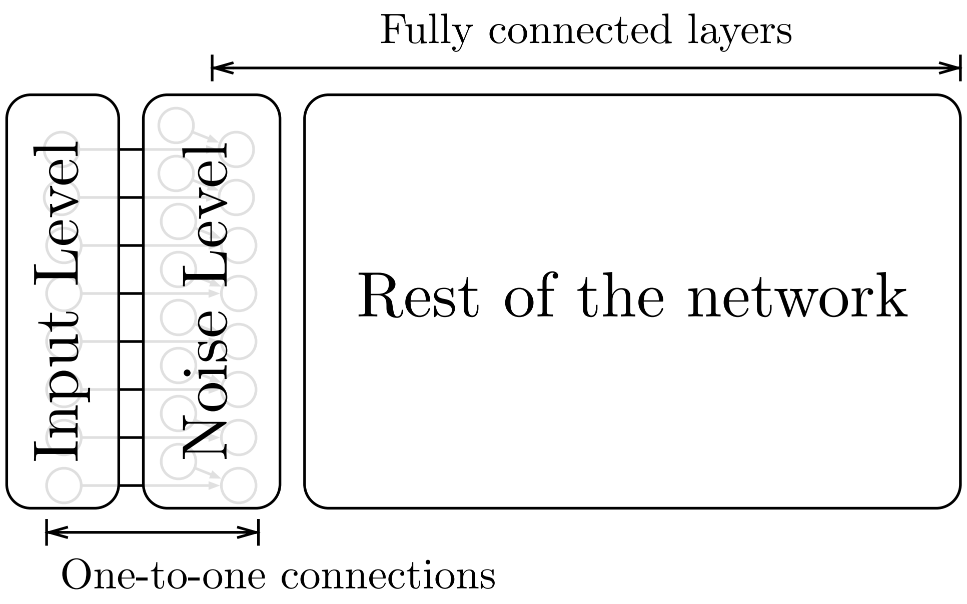


Figure 5.2: The general structure of the networks which include a noise layer. The sub-networks $f^y_{\theta_y}$ and $f^s_{\theta_s}$ are omitted in this figure.

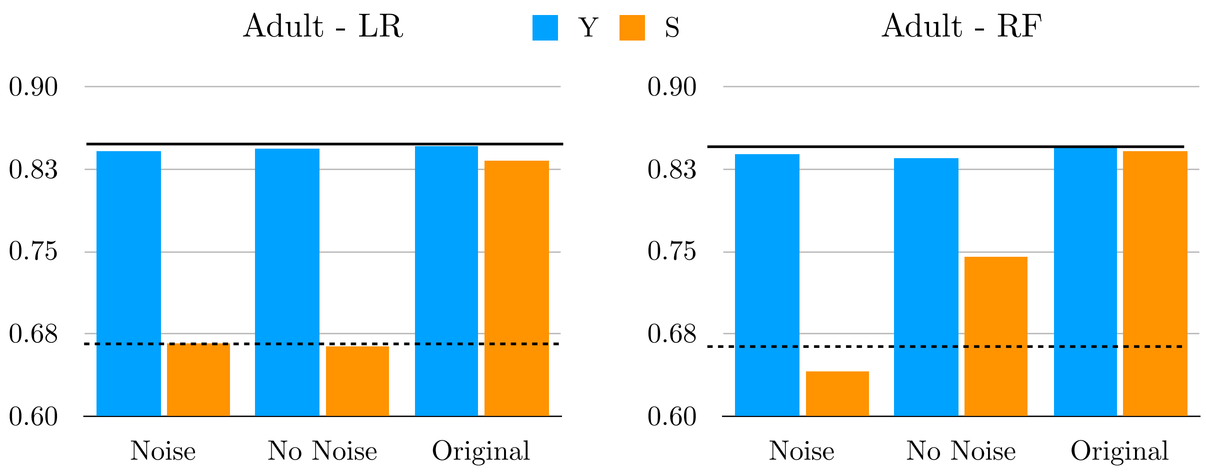


Figure 5.3: Results for logistic regression (LR) and random forest (RF) algorithms when trained on the extracted representations on the Adult dataset. The continuous line refers to the accuracy obtained on $y$ by the classifier on the original representations. The dotted line refers to the accuracy obtained on $s$ on the original representations.

evaluation here is performed by estimating classifier performance on both $(\hat{x}, y)$ and $(\hat{x}, s)$, comparing it with classifier performance on the original representations, i.e. the datasets $(x, y)$ and $(x, s)$. The ideal result here is to achieve accuracy on $(\hat{x}, y)$ that is just as good as the one on $(x, y)$ while approaching the random guess performance (i.e. the majority class) on $(\hat{x}, s)$. This latter result would imply that different values of the sensitive attribute $s$ are indistinguishable in the learned representations.

We observe that our methodology leads to representations which are both fair and discriminative on all datasets, as visible in Figures 5.3 through 5.6. In these figures, the solid line refers to the accuracy achievable on the original

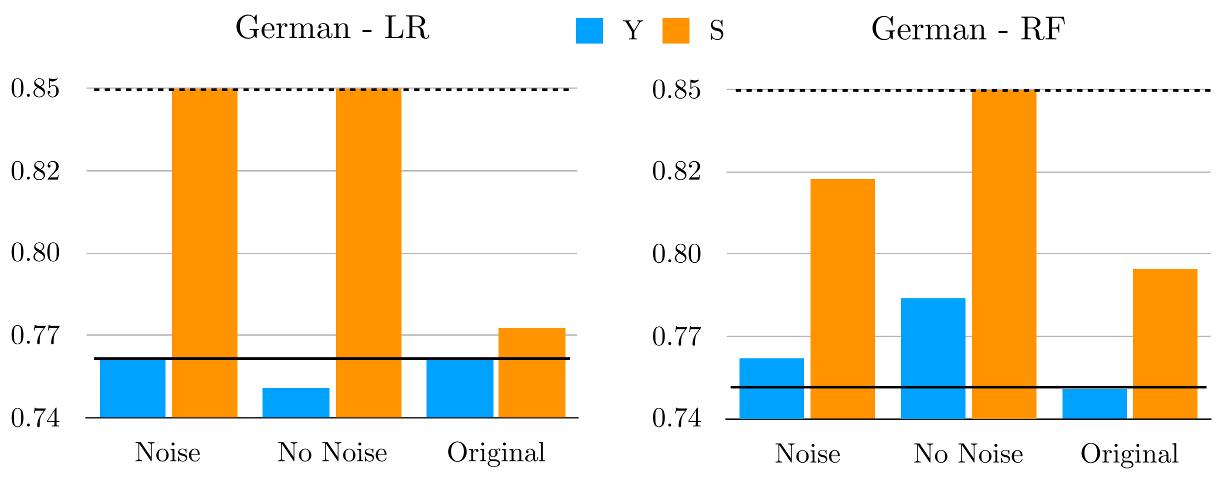


Figure 5.4: Results for logistic regression (LR) and random forest (RF) algorithms when trained on the extracted representations on the German dataset. The continuous line refers to the accuracy obtained on $y$ by the classifier on the original representations. The dotted line refers to the accuracy obtained on $s$ on the original representations.

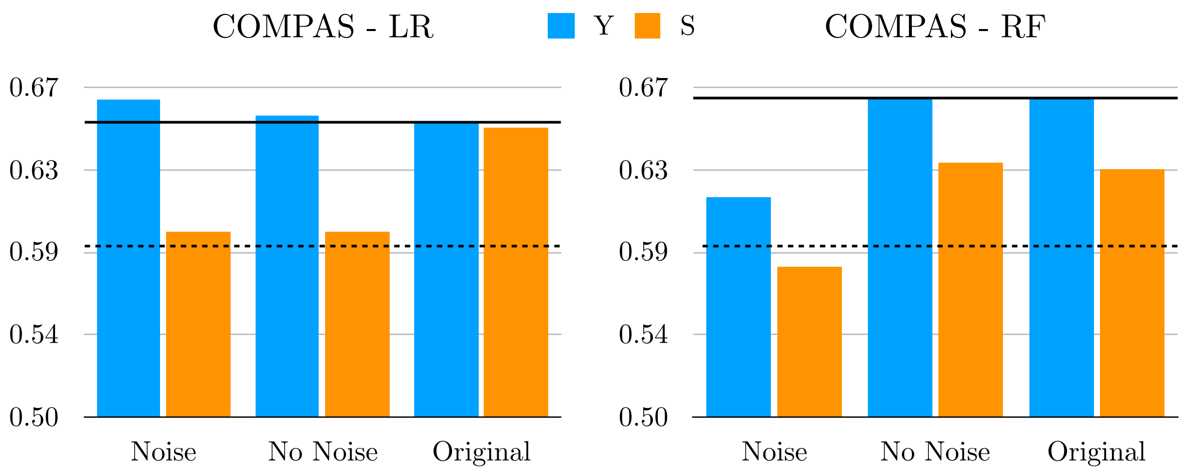


Figure 5.5: Results for logistic regression (LR) and random forest (RF) algorithms when trained on the extracted representations on the COMPAS dataset. The continuous line refers to the accuracy obtained on $y$ by the classifier on the original representations. The dotted line refers to the accuracy obtained on $s$ on the original representations.

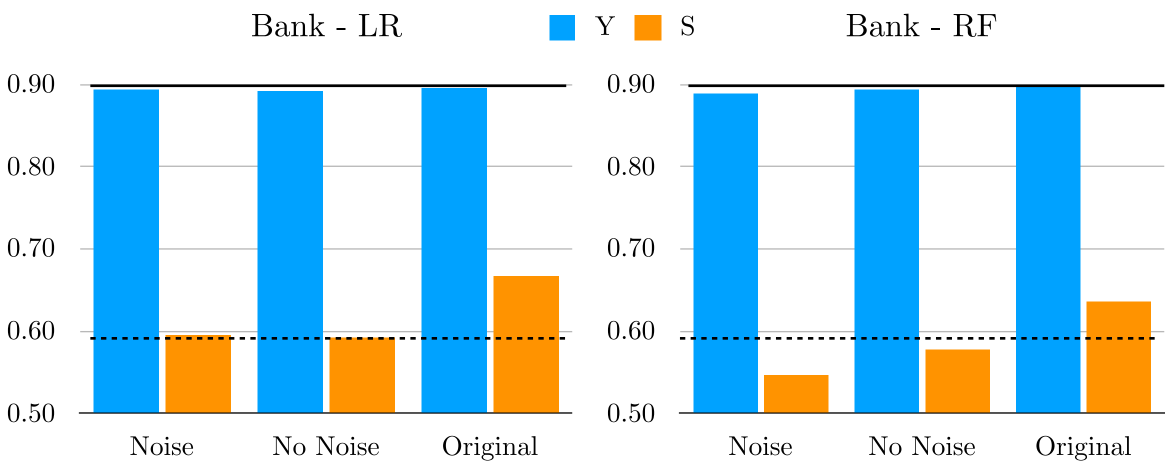


Figure 5.6: Results for logistic regression (LR) and random forest (RF) algorithms when trained on the extracted representations on the COMPAS dataset. The continuous line refers to the accuracy obtained on $y$ by the classifier on the original representations. The dotted line refers to the accuracy obtained on $s$ on the original representations.

representation for the data when predicting $y$, while the dotted line indicates the random chance accuracy for $s$. Thus, a classifier trained to predict $s$ with a perfectly fair representation would display performance which matches the dotted line, while the performance of a classifier trained to predict $y$ with a representation which has not lost any of its discriminative information would match the solid line.

On the Adult dataset (Figure 5.3), the representations learned by the network which employs a noise module have achieved true invariance w.r.t. $s$, being close or slightly below the random chance baseline. On the other hand, the standard adversarial strategy is unable to account for all the correlations between the attributes and the sensitive attribute, as the random forest classifier is able to predict $s$ with some accuracy, albeit lower than on the original representation. Experiments on the COMPAS data provide similar insights. Including the noise module is critical in achieving fair representations, as the random forest accuracy on $s$ is even higher than on the original data when employing the representations learned from the standard adversarial strategy. The German dataset displays high levels of class imbalance which lead to non-discriminative representations by all strategies we tested; however, fairness was still preserved.

## 5.2 Fair Ranking via Debiasing

The problem of *learning to rank* is a fundamental application in information retrieval. In this problem, a list of $n$ documents needs to be sorted based on relevance w.r.t. a query $q$. Ranking models might also be employed in situations where they have a direct impact on individuals' well-being, such as ranking candidates for admission to higher education or giving out scholarships. Models that address the *learning to rank* problem, in which a list of $n$ documents needs to be sorted based on relevance to a query, fall into three broad categories according to whether the objective function is computed by considering one, two or the whole list of documents at a time during training. The first approach is called *pointwise* and is analogous to classifying each document [CGD92; LWB08] in the sense that instead of comparing documents in a list, a score is predicted on each query-document pair, indicating the single document's relevance to the query. In the *pairwise* approach a model tries to determine the more relevant document out of two for the given query [Bur+05; Fri00]. The last approach is called *listwise*, in which the whole list is used to compute the cost during training [Cao+07a; XL07].

It is possible to extend many classification algorithms to the ranking problem, such as decision trees [Fri00], support vector machines [Cao+06], artificial neural networks [Cao+07a] and ensemble boosting [Wu+10].

As far as neural networks are concerned, the issues in employing these models in ranking problems are of opacity and fairness. While models such as ListNet [Cao+07a] and RankNet [Bur+05] are state of the art on general tasks, it is hard to make sense of their decisions, a problematic issue when people's well being may be impacted. Therefore, research on fair classification has been extended in recent years to the problem of ranking *fairly*.

In fair ranking one wants to find a quasiorder of documents according to their relevance while guaranteeing some notion of fairness with respect to a sensitive attribute. Let us define a dataset $D = \{(q_i, x_i, s_i, y_i), i \in \{1...N\}\}$, where $q_i$ are the queries, $x_i \in \mathbb{R}^m$ are vectors describing the non-sensitive attributes of the documents, $s_i \in \mathbb{R}^n$ are vectors describing sensitive attributes, and $y_i$ represents each document's relevance for the query. The main motivation for exploring the task of ranking fairly is ensuring that individuals belonging to protected groups are not relegated to lower ranking positions [Yan+18] because of past or present human biases which may be reflected in training data.

In this context, my contribution is to devise a pairwise neural network which employs explicit mechanisms for fairness [Cer+20]. This network has been developed as an extension of the state of the art model DirectRanker [Köp+19] by Köppel et al. This contribution has been published in the IEEE Data Science and Advanced Analytics 2020 Conference in collaboration with Dr. Marius Köppel, Dr. Alexander Segner, Prof. Dr. Stefan Kramer and Prof. Roberto Esposito.

### 5.2.1 The DirectRanker Architecture

The DirectRanker framework enforces a number of properties which enable it to learn a total quasiorder on a wide variety of feature spaces. Given a quasiorder $\succeq$ defined on pairs of documents $(x_i, x_j, x_k)$, the following conditions need to be satisfied:

- Reflexivity: $x_i \succeq x_i$
- Antisymmetry: $x_i \not\succeq x_j \rightarrow x_j \succeq x_i$
- Transitivity: $(x_i \succeq x_j \succeq x_k) \rightarrow x_i \succeq x_k$

These conditions might be implemented by a pairwise ranking function $f : (\mathcal{X}, \mathcal{X}) \rightarrow \mathcal{R}$ by defining it as follows:

$$x_i \succeq x_j \iff f(x_i, x_j) \geq 0 \tag{5.2}$$

Then, the aforementioned conditions are imposed on the ranking function $f$ as follows:

- Reflexivity: $f(x_i, x_i) = 0$
- Antisymmetry: $f(x_i, x_j) = -f(x_j, x_i)$
- Transitivity: $(f(x_i, x_j) \geq 0 \ \land \ f(x_j, x_t) \geq 0) \rightarrow f(x_i, x_t) \geq 0$

These properties can be seen as structural constraints on a neural network architecture. Rigutini et al. [Rig+08] first discussed how to impose antisymmetry on a network by means of an antisymmetric activation function and the removal of the bias term in the last layer; reflexivity follows from a siamese network structure which extracts features from two different documents and subtracts them. The DirectRanker model is inspired from these insights and is defined as follows:

$$f(x_i, x_j) = \sigma(g(x_1) - g(x_2)) \tag{5.3}$$

Where $g$ is a neural network which extracts features from the documents and $\sigma$ is an antisymmetric, sign-conserving function such as the tangent hyperbolic. A sketch of the complete architecture can be found in Figure 5.7.

To further constrain the DirectRanker architecture for fairness, in our contribution we propose two different mechanisms which are shown to be useful in obtaining fair results.

### 5.2.2 Adversarial Fair DirectRanker

As described in Section 5.1, one possible technique to obtain invariant representations is to employ a sub-network which is trained to predict the sensitive

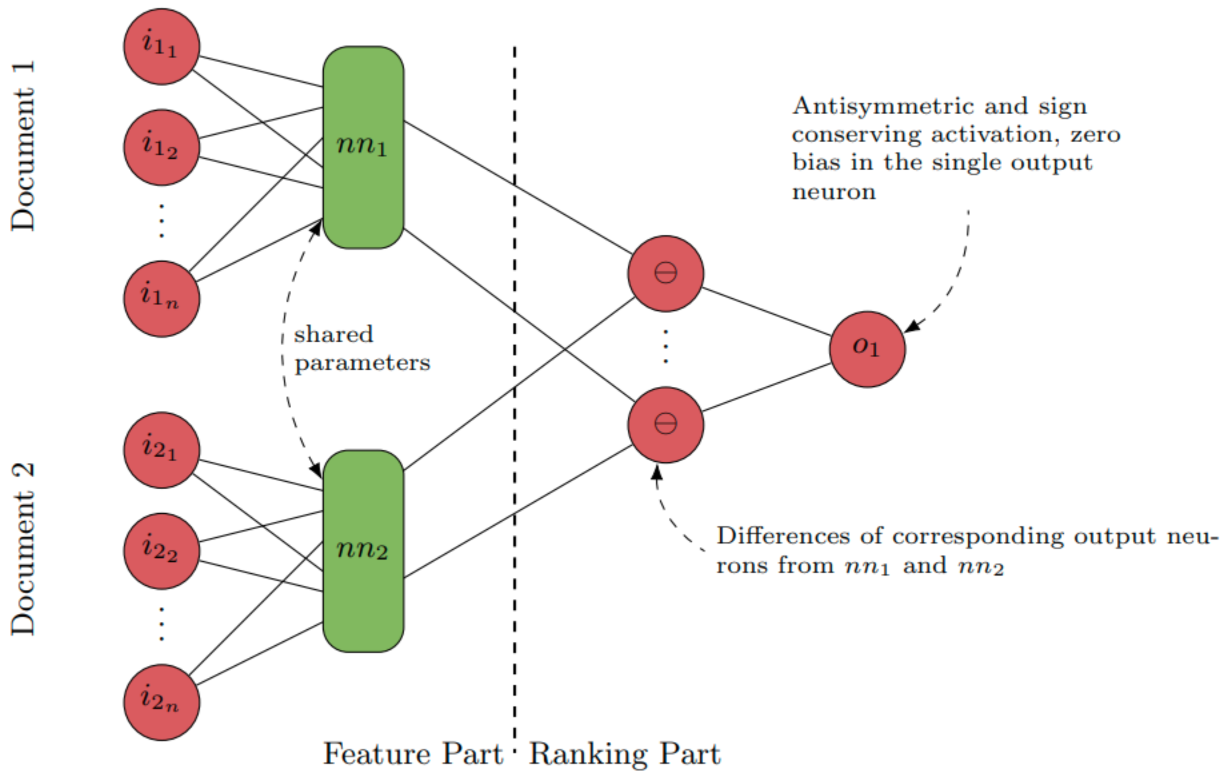

Document 1
Document 2
$i_{1_1}$
$i_{1_2}$
$i_{1_n}$
$i_{2_1}$
$i_{2_2}$
$i_{2_n}$
$nn_1$
$nn_2$
shared parameters
$\ominus$
$o_1$
Antisymmetric and sign conserving activation, zero bias in the single output neuron
Differences of corresponding output neurons from $nn_1$ and $nn_2$
Feature Part
Ranking Part

Figure 5.7: The DirectRanker model by Köppel et al.. Figure extracted from [Köp+19] and used with permission from the author.

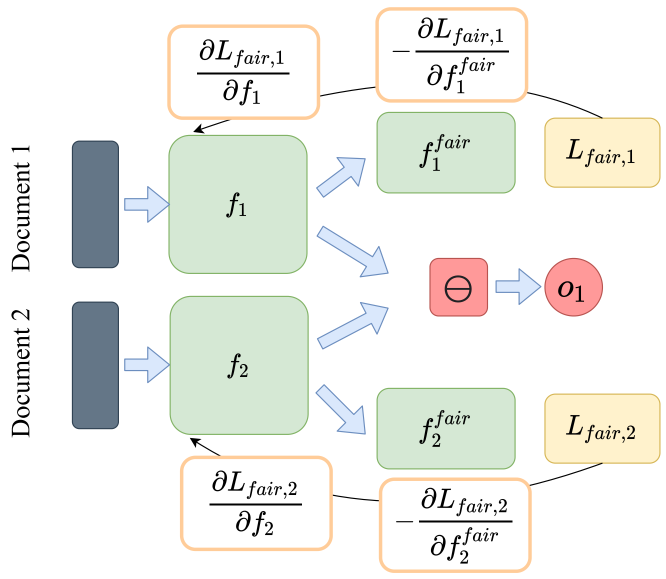

$\frac{\partial L_{fair,1}}{\partial f_1}$
$-\frac{\partial L_{fair,1}}{\partial f_1^{fair}}$
Document 1
Document 2
$f_1$
$f_2$
$f_1^{fair}$
$L_{fair,1}$
$\ominus$
$o_1$
$f_2^{fair}$
$L_{fair,2}$
$\frac{\partial L_{fair,2}}{\partial f_2}$
$-\frac{\partial L_{fair,2}}{\partial f_2^{fair}}$

Figure 5.8: The Adversarial Fair DirectRanker architecture. Two sub-networks, one for each of the feature extraction siamese networks, are trained to predict the sensitive attribute.

attribute. The gradient of this network can then be reversed, leading to a min-max problem of optimizing for $y$ and learning invariant representations w.r.t. $s$.

Here we employ the same concept by augmenting the DirectRanker architecture via two sub-networks $f_1^{fair}$ and $f_2^{fair}$ which are trained to predict the sensitive attribute and back-propagate their inverted gradient into the feature extraction part of the network. A diagram for this network may be found in Figure 5.8

The complete loss function for this model is as follows:

$$L(\Delta y, x_1, x_2, s_1, s_2) = L_{rank}(\Delta y, x_1, x_2) + \lambda \sum_{i=1}^{2} L_{fair,i}(s_i, x_i) \tag{5.4}$$

$$L_{rank} = (\Delta y - o_1(x_1, x_2))^2 \tag{5.5}$$

$$L_{fair,i} = -s\ log(f_i^{fair}(x)) - (1-s)\ log\ (1 - f_i^{fair}(x)) \tag{5.6}$$

In which $\Delta y = \mathbb{I}_{y1 \geq y2}$. Following the strategy outlined by Köppel et al., we ensure that the document with higher relevance is always $x_1$. Therefore, $\Delta y$ is set to be always 1. In Equation 5.6 $s$ is assumed to be binary for simplicity, but the cross entropy loss might be employed also in the more general case where $s$ is not binary. The $\lambda$ hyperparameter regulates the relevance-fairness tradeoff and may be selected via the usage of a validation set.

### 5.2.3 Flipped Fair DirectRanker

The Flipped Fair DirectRanker builds on the previous strategy by reversing a *ranking* gradient instead of a classification one. That is, another output neuron $o_{fair}$ is added to the network. This neuron does not have a bias term and employs a sign-conserving, asymmetric activation function. The idea here is to rank documents according to their sensitive attribute; then, by inverting the gradient, the model is optimized to be agnostic to the difference between the sensitive attribute of two input documents. A diagram of this architecture can be found in Figure 5.9.

The objective function for this network is defined as follows:

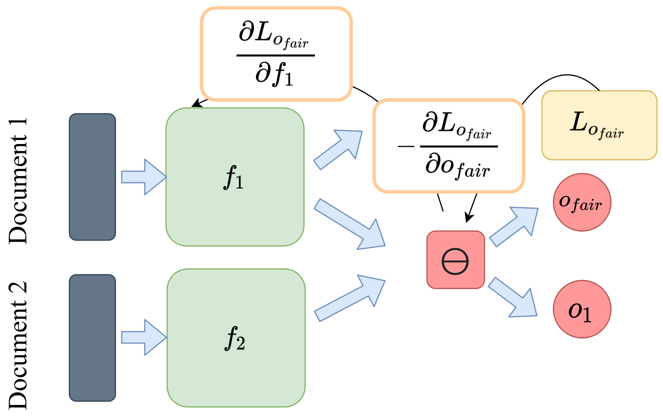

$\frac{\partial L_{o_{fair}}}{\partial f_1}$
$-\frac{\partial L_{o_{fair}}}{\partial o_{fair}}$
$L_{o_{fair}}$
Document 1
Document 2
$f_1$
$f_2$
$\ominus$
$o_{fair}$
$o_1$

Figure 5.9: The Flipped Fair DirectRanker architecture. Another output neuron $o_{fair}$ is added to the architecture. The neuron is employed to rank on values of the sensitive attribute, but its gradient is inverted.

$$L(\Delta y, \Delta s, x_1, x_2) = L_{rank}(\Delta y, x_1, x_2) + \lambda L_{fair}(\Delta s, x_1, x_2) \tag{5.7}$$

$$L_{rank} = (\Delta y - o_1(x_1, x_2))^2 \tag{5.8}$$

$$L_{fair,i} = (\Delta s - o_{fair}(x_1, x_2))^2 \tag{5.9}$$

Note that the ranking objectives for the Flipped Fair DirectRanker and the Adversarial Fair DirectRanker are the same (cfr. Equations 5.5 and 5.8). Here, $\Delta s$ is simply defined as $\mathbb{I}_{s_1 > s_2}$.

### 5.2.4 Experiments and Results

In the following we evaluate our models on ranking datasets commonly used in the fairness literature. To encourage fairness in the model's decisions, we employ the aforementioned two different mechanisms (Sections 5.2.2 and 5.2.3), which we evaluate separately for relevance and fairness by using standard metrics. We also show results for models which include both the mechanisms. For the relevance task, we use the commonly used nDCG@$k$ and the AUROC, referred to here as AUC. The fairness of the models is evaluated via the rND metric (Section 3.1.2) and explicitly by the accuracy obtained when predicting the sensitive attribute from the representations they learn. We also report the group-dependent pairwise accuracies (Section 3.1.2), which have been recently developed [Nar+20]. We explored a number of hyperparameter combinations, which we report in Table 5.1. Our evaluations are twofold, as we assess both the ranking models and their *extracted representations*, i.e. the features learned by the feature extraction layers. This can be done by training a supervised model on the aforementioned representations. This is a standard way to evaluate fair classifiers (see e.g. Zemel et al. [Zem+13] and Section 5.1) and to understand how much information about $s$ has been removed by the model, as high values for fairness can also be achieved by a weak ranker taking random or quasi-random decisions from biased data. We transpose this setup to the fair ranking paradigm by training both linear and non-linear classifiers and rankers (logistic regression and random forest models, respectively). We then evaluate their invariance to the sensitive attribute and the fairness of the ranked outputs. As mentioned previously, this experimental setup can also simulate a "separations of concerns" regulatory scenario [MOW17]. In this setting, users intending to train models which directly impact individuals only have access to representations where information about sensitive data has been purged. This enables any downstream model to be fair by design, but poses the additional challenge of having to remove information from the data.

### Relevance Metrics

In the field of learning to rank, a commonly used measure to evaluate the performance of a model is the normalized discounted cumulative gain of top-$k$ documents retrieved (NDCG@$k$). This metric is based on the discounted cumulative gain of top-$k$ documents (DCG@$k$):

$$\mathrm{DCG@}k = \sum_{i=1}^{k} \frac{2^{r(d_i)} - 1}{log_2(i+1)}, \tag{5.10}$$

where $d_1, d_2, ..., d_n$ is the list of documents sorted by the model with respect to a single query and $r(d_i)$ is the relevance label of document $d_i$. The NDCG@$k$ can be computed by dividing the DCG@$k$ by the ideal (maximum) discounted cumulative gain of top-$k$ documents retrieved (IDCG@$k$), i.e. the DCG@$k$ for a perfectly sorted list of documents is defined as NDCG@$k = \frac{\mathrm{DCG@}k}{\mathrm{IDCG@}k}$. We report values for NDCG@500.

### Fairness Metrics

To evaluate group fairness in the whole output list, we employ the rND metric introduced by Ke and Stoyanovitch [YS17] and here analyzed in Section 3.1.2. As previously mentioned, this metric may be employed to analyze the disparate treatment in a ranker. The Group-dependent Pairwise accuracy has also been employed to instead analyze the disparate mistreatment of our rankers (see Section 3.1.2).

### Datasets

Our experimental evaluation is focused on datasets commonly employed in the fairness literature, such as Adult [Koh96], COMPAS [Ang+16], and Law Students [ZC19]. We also employed the Wiki Talk Page Comments dataset to enable a comparison with a recently developed neural-based constrained optimization method [Nar+20]. Details on these datasets may be found in Section 3.3. One thing of note is that previous literature employing the Wiki Talk Page dataset has employed a convolutional neural network [Nar+20]. Our approach instead is to first preprocess the comments to generate a word representation by using a pre-trained model [Mik+13]. The obtained representation is then fed into our models.

### 5.2.5 Grid Search

We performed a grid search to find the best hyperparameters and architecture for our models. As is often observed in the fairness literature, removing information about the sensitive attribute often leads to decreased accuracy or relevance of the model. The same phenomenon has been observed on representations extracted from fair neural models [Lou+15; CEL20]. To the end of obtaining models and representations which are both fair and relevant, we employed a metric which combines fairness and relevance for both the models and the representations extracted from them. Relevance and fairness were weighted equally. We split all the datasets with a 60/20/20 train/validation/test ratio. We then selected the models having the highest value for our metric on the validation set and in the following report their performance on the test set. The hyperparameter set we optimized may be found in Table 5.1.

Furthermore, we tested models that had a noise module as the first layer of their architecture. For an implementation of the models and the experimental setup, see *https://zenodo.org/record/3889006*.

Table 5.1: Overview of the hyperparameters included in the grid search for the fair ranking strategy

| parameter | values |
|---|---|
| activation function | tanh |
| #layers of $nn_{1/2}$ | 2 |
| #neurons per layer in $nn_{1/2}$ | 5, 10, 20, ..., 100 |
| #layers of $nn_{bias_1/bias_2}$ | 2 |
| #neurons per layer in $nn_{bias_1/bias_2}$ | 2, 10, 20, ..., 50 |
| training steps | 0.5k, 1k, 1.5k, ..., 10k |
| $\lambda$ | 0.1, 1, 2, 5, 10, 100 |

### 5.2.6 Experiments Results

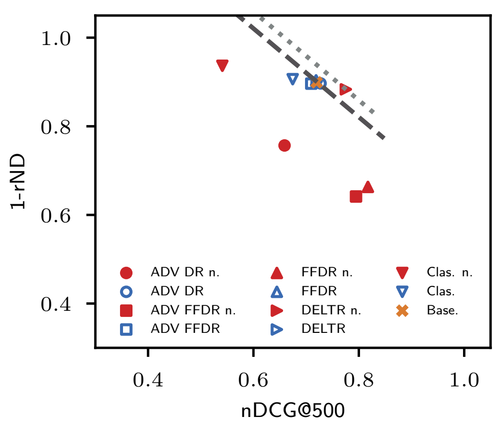
In this section we present the experimental results. First, we look at the results for the different rankers on various datasets. Furthermore, we compare the results of the external rankers and classifiers trained on the representations extracted by the same models. On the *Wiki* dataset, we also report the AUC and the GPA metrics to enable a comparison to related work that phrases fairness definitions as constrained optimization problems [Nar+20] (*Con. Opti.* in the figures.).

The comparison methods on all other datasets include: a fair listwise ranking model [ZC19] (*DELTR*) that we also augmented with a noise module [CEL20] (*DELTR n.*), a debiasing neural classifier [Gan+16; CEL20] (*Clas* and *Clas n.* if including a noise module), and an "unfair" baseline (*Base.*), which is the base DirectRanker model with $\gamma = 0$. Models employing the adversarial mechanism of Section 5.2.2 are included with the name "ADV DR", while the ones which include the "fair flipping" mechanism of Section 5.2.3 are named "FFDR". We also employed a model that combines both the fairness mechanisms introduced in this work (*ADV FFDR, ADV FFDR n.*). These models use both the loss functions introduced in Sections 5.2.2 and 5.2.3.

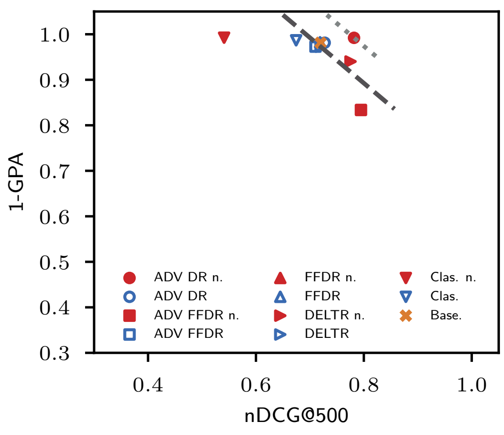
#### Model results

In the following we will discuss our results as far as the relevance (nDCG or AUC) and fairness (rND, Section 3.1.2, or GPA, Section 3.1.2) of the models themselves are concerned. These results can be found in Figs. 3 through 5. Note that for ease of reading, the rND and GPA metric is reported as 1-rND and 1-GPA, respectively. In all the figures. we plot the optimal line representing the model which finds the best trade-off, i.e. the smallest value of $\|(1,1) - (1-m,n)\|_1$ with $(m,n) \in \{(\text{rND},\text{nDCG}),(\text{GPA},\text{AUC})\}$. The line for the comparison models is drawn dashed, while the one for all models (including ours) is dotted. For all datasets, we find that members of the proposed method family push the trade-off closer to the best possible model regarding both of the considered fairness metrics.

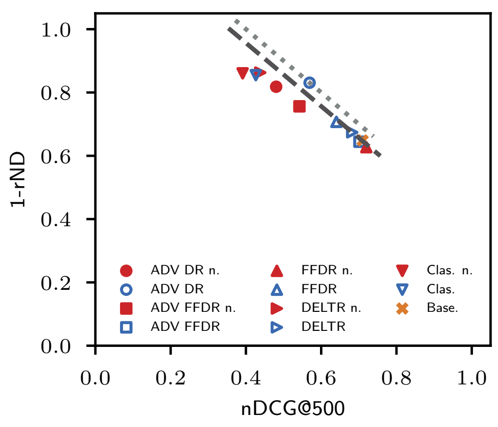
While no single model is able to outperform the comparisons on both nDCG and rND/GPA, our models are able to find strong trade-offs between relevance and fairness in all employed datasets. Overall, we found that models including the Adversarial fairness mechanism (ADV DR, ADV FFDR) performed the best. On Adult (Figs. 5.10(a) and 5.10(b)), the ADV DR model is just as fair as DELTR, while also producing rankings which are slightly more relevant. On COMPAS (Figs. 5.10(c) and 5.10(d)), our ADV DR model produces rankings which are competitive in fairness with respect to DELTR n. and the Debias Classifier. However, the output list has higher nDCG. The law dataset with gender as a sensitive attribute (Figs. 5.11(a) and 5.11(b)) provides similar insights, where, however, our best performing



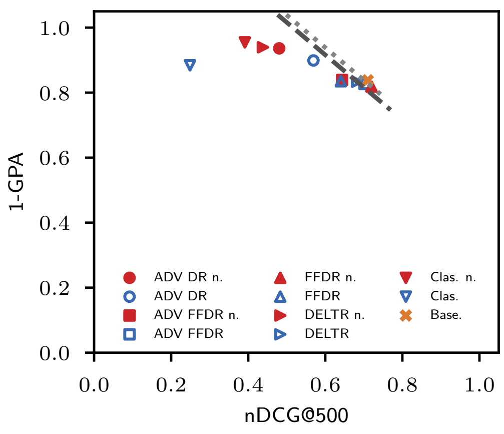
(a) Adult rND/nDCG results



(b) Adult GPA/nDCG results

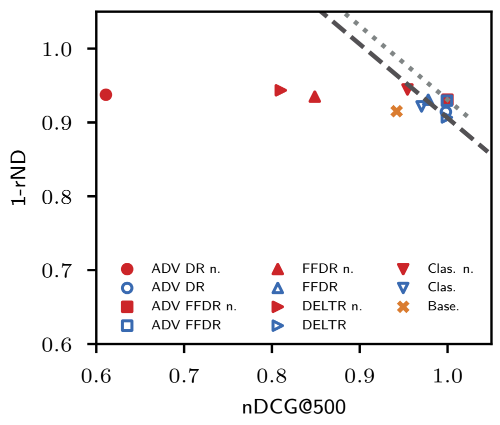


(c) COMPAS rND/nDCG results

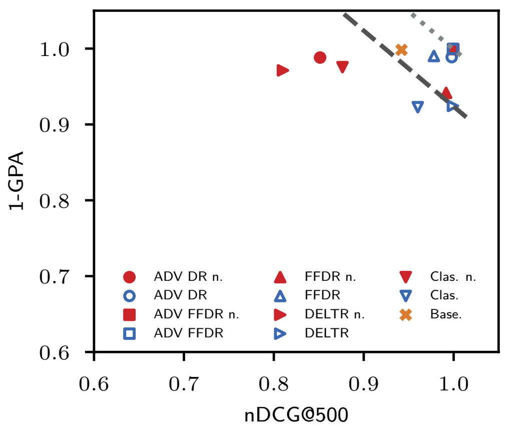


(d) COMPAS GPA/nDCG results

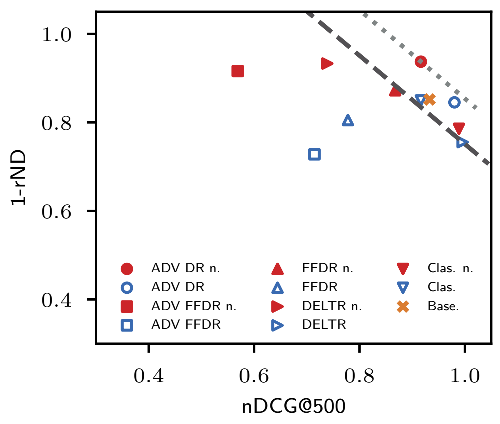
Figure 5.10: Results for the *ADULT* and *COMPAS* datasets for the ranker outputs. Models marked with “n.” such as ADV DR n. included a noise module in their architecture. The lines represent balanced fairness/relevance trade-offs. The dashed line represents all points with equal trade-off as the best performing comparison model, while the dotted line also takes into consideration the models we contribute.



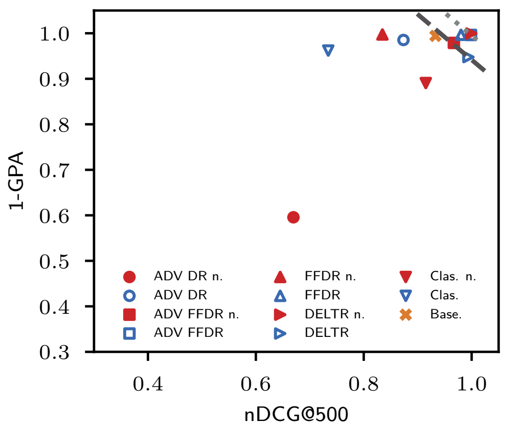
(a) Law-gender rND/nDCG results



(b) Law-gender GPA/nDCG results



(c) Law-race rND/nDCG results



(d) Law-race GPA/nDCG results

Figure 5.11: Results for the *Law Students* datasets for the ranker outputs. Models marked with “n.” such as ADV DR n. included a noise module in their architecture. The lines represent balanced fairness/relevance trade-offs. The dashed line represents all points with equal trade-off as the best performing comparison model, while the dotted line also takes into consideration the models we contribute.

model employed both the mechanisms from this paper (ADV FF DR). Once again, the Debias Classifier found rankings which are fair but at the cost of slightly lowered relevance, possibly as an effect of not employing a specialized ranking loss. When considering race as the sensitive attribute (Figs. 5.11(c) and 5.11(d)), the ADV DR n. is able to retrieve the fairest rankings, while still being competitive on nDCG.

When considering the AUC/GPA trade-off on the *Wiki* dataset (Fig. 5.12(c)), ADV DR and FFDR have an identical performance, which is, however, surpassed by the constrained optimization model. As for the nDCG/rND trade-off (Fig. 5.12(a)), the ADV DR n. is once again able to find rankings which compare favorably to the others in terms of fairness, however, also having severely reduced nDCG. On this dataset, the ADV FFDR n. struck an impressive trade-off, which is about as fair as the fairest methods on the dataset (DELTR, Clas.), while also having visibly higher nDCG. Similar observations can be made in the case of the nDCG/GPA tradeoff (Figure 5.12(b)). In general, it is worthwhile to note how the DELTR model obtained fairer rankings on two of the datasets (COMPAS, law-race) when also employing a noise module.

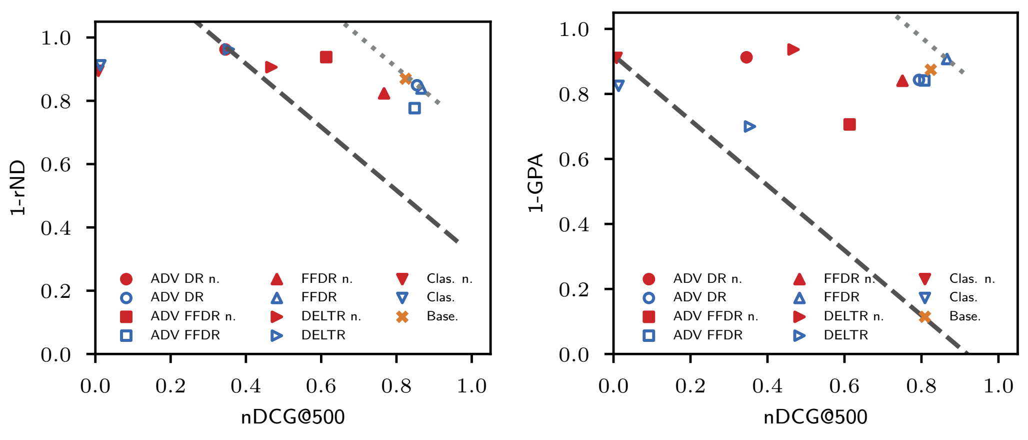
### Representation results

We report results for external rankers and classifiers which have been trained on representations extracted from the employed models in Table 5.2. We trained both linear (logistic regression) and non-linear (random forest) models and report three different metrics. To evaluate the invariance of the representation with respect to the sensitive attribute, we report classifier accuracy as the absolute distance from random guess (the majority class ratio in the dataset), which is shown as ADRG in Table 5.2. Similar analyses are common in fair classification [CEL20; Xie+17; Lou+15]. We transfer this analysis to the fair ranking paradigm by training rankers on the aforemen-

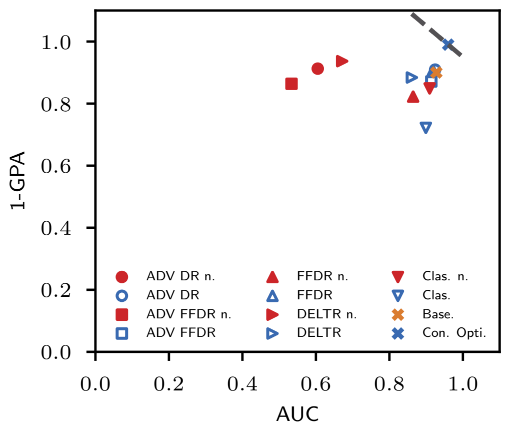
Table 5.2: Performance of the representations rankers for all models on different fair datasets. The metrics employed are, from left to right, absolute difference to random guess, 1-rND, and the NDCG@500. Values are marked bold if they are the highest ones of the metric. Models marked with `n.` employed a noise module in their first layer.

| | COMPAS | | | Law-gender | | | Adult | | |
|---|---|---|---|---|---|---|---|---|---|
| Models | ADRG | 1-rND | nDCG | ADRG | 1-rND | nDCG | ADRG | 1-rND | nDCG |
| ADV DR n. | 0.03 | 0.95 | 0.69 | 0.04 | 0.93 | **1.00** | 0.13 | 0.93 | **0.85** |
| ADV DR | 0.02 | 0.87 | 0.48 | 0.04 | 0.92 | 0.77 | **0.01** | 0.95 | 0.68 |
| ADV FFDR n. | 0.03 | 0.95 | 0.63 | 0.04 | 0.87 | 0.88 | **0.01** | **0.98** | 0.75 |
| ADV FFDR | 0.03 | 0.86 | 0.56 | 0.04 | 0.96 | 0.92 | **0.01** | 0.95 | 0.71 |
| FFDR n. | 0.07 | 0.88 | 0.66 | 0.04 | 0.92 | 0.81 | **0.01** | 0.95 | 0.77 |
| FFDR | 0.03 | **0.96** | 0.35 | 0.07 | 0.95 | 0.89 | **0.01** | 0.96 | 0.73 |
| DELTR n. | 0.03 | 0.95 | **0.72** | 0.03 | 0.92 | 0.98 | 0.04 | 0.90 | 0.83 |
| DELTR | **0.01** | 0.90 | 0.35 | **0.02** | 0.94 | 0.99 | **0.01** | 0.95 | 0.73 |
| Clas. n. | 0.04 | 0.86 | 0.45 | 0.04 | 0.91 | 0.97 | **0.01** | **0.98** | 0.56 |
| Clas. | 0.04 | 0.90 | 0.40 | 0.04 | 0.93 | 0.95 | **0.01** | 0.91 | 0.68 |
| Base. | 0.02 | 0.77 | 0.30 | 0.05 | **0.97** | 0.93 | **0.01** | 0.95 | 0.71 |

| | Law-race | | | Wiki | | |
|---|---|---|---|---|---|---|
| Models | ADRG | 1-rND | nDCG | ADRG | 1-rND | nDCG |
| ADV DR n. | **0.01** | 0.92 | 0.89 | 0.01 | 0.93 | **0.58** |
| ADV DR | 0.03 | 0.90 | 0.87 | **0.00** | 0.94 | 0.51 |
| ADV FFDR n. | 0.06 | **0.95** | 0.95 | 0.01 | 0.93 | 0.17 |
| ADV FFDR | **0.01** | 0.86 | 0.79 | **0.00** | 0.92 | 0.04 |
| FFDR n. | **0.01** | 0.94 | **1.00** | **0.00** | 0.94 | 0.06 |
| FFDR | 0.06 | 0.90 | 0.98 | **0.00** | 0.90 | 0.08 |
| DELTR n. | 0.02 | **0.95** | 0.81 | **0.00** | **0.95** | 0.12 |
| DELTR | 0.03 | 0.82 | 0.89 | **0.00** | 0.89 | 0.55 |
| Clas. n. | 0.03 | 0.91 | 0.72 | 0.40 | 0.93 | 0.50 |
| Clas. | 0.20 | 0.93 | 0.68 | 0.02 | 0.91 | 0.55 |
| Base. | **0.01** | 0.90 | 0.94 | **0.00** | 0.92 | 0.22 |



(a) Wiki rND/nDCG results (b) Wiki GPA/nDCG results



(c) Wiki AUC/GPA results

Figure 5.12: Results for the *Wiki* datasets for the rankers. Models marked with n. included a noise conditioning layer. In Fig. 5.12(c), the values for Con. Opti. are taken from the original publication [Nar+20]. The lines represent balanced fairness/relevance trade-offs. The dashed line represents all points with equal trade-off as the best performing comparison model, while the dotted line also takes into consideration the models we contribute.

tioned representations. Therefore, we report the nDCG and 1-rND values for these rankers. We report the most unfair (ADRG, 1-rND) and the most relevant (nDCG) values for the representations extracted from each model.

Experiments on the extracted representations confirm the insights derived from the model analysis, where models employing the Adversarial mechanism introduced were the best performing. On Adult, ADV DR and ADV FFDR are once again strong performers in all metrics, invariance included. ADV DR n. is a strong performer on COMPAS, where however DELTR n. is able to achieve better nDCG results. On law-gender ADV DR n. and ADV FF DR obtain impressive results on nDCG and rND, respectively, while still having solid performance. When instead using race as the sensitive attribute, ADV FF DR n. learns representations which have the lowest rND value, while FF DR n. displays the best nDCG with a very minor loss in rND. On *Wiki*, the DELTR n. model performs the best on the rND metric, however, computing outputs which are hard to sort in a relevant way. The ADV DR n. model, in comparison, has the highest value for nDCG and competitive values for rND.

## 5.3 Towards Interpretable Fair Representation Learning

In the previous sections, we have seen how to leverage an invariant representation learning algorithm for fairness. While the contributions presented so far can learn fair models, they still have an issue on terms of *opacity*. That is, it is hard to understand why a particular decision has been made. More in general, these methods project the original data space $\mathcal{X}$ into a latent space $\mathcal{Z}$ whose dimensions are incomprehensible to humans, as they are non-linear combinations of the original features. This is a noteworthy problem especially in the context of the "right to an explanation" as required in the EU by the GDPR, Recital 71. Therefore, these methodologies might be inapplicable in the real world.

In this context, in this section I will present a new fair representation learning framework which learns *feature corrections* instead of an entirely new space of opaque parameters. This methodology has been published as a workshop paper at the International Conference for Learning Representations 2021. In practice, this is akin to a pre-processing technique which changes the original features so to balance them between individuals belonging to different groups. This guarantees a “right to an explanation” in the sense that it is always possible to extract the “fair correction” that has been applied to the data belonging to each individual. Furthermore, as the correction is computed via neural networks, our framework still enjoys all the benefits of the universal approximation theorems (see [Cyb89]) and may therefore compute any debiasing function. This framework is flexible, as it imposes only architectural constraints on the neural network without impacting the training objective: therefore, all neural debiasing methodologies may be extended so to belong in the framework.

Commonly, the learning of fair representations is achieved by learning a new feature space $\mathcal{Z}$ starting from the input space $\mathcal{X}$. To this end, a parameterized function $f_\theta(x)$ is trained on the data and some debiasing component which looks at the sensitive data $s$ is included. After training, debiased data is available by simply applying the learned function $z = f_\theta(x)$. Any off-the-shelf model can then be employed on the debiased vectors. Various authors have investigated techniques based on different base algorithms.

The issue with the aforementioned strategy is one of interpretability. While it is possible to guarantee *invariance to the sensitive attribute* – with much effort – by training classifiers on the debiased data to predict the sensitive attribute, it is unknown what each of the dimensions of $\mathcal{Z}$ represent. Depending on the relevant legislation, this can severely limit the applicability of fair representation learning techniques in industry. Our proposal is to mitigate this issue by instead *learning fair corrections* for each of the dimensions in $\mathcal{X}$. Fair corrections are then added to the original vectors so that the semantics of the algorithm are as clear as possible. For each feature, an individual will have a clear penality or bonus depending on the sign of the correction. Thus, we propose to learn the latent feature space $\mathcal{Z}$ by learning fair corrections $w$: $w = f_\theta(x)$ and $z = w + x$.

It is very practical to modify existing neural network architectures so that they can belong in the aforementioned framework. While there are some architectural constraints that have to be enforced, the learning objectives and training algorithms may be left unchanged. The main restriction is that only "autoencoder-shaped" architectures may belong in our framework. Plainly put, the depth of the network is still a free parameter, just as the number of neurons in each hidden layer. However, to make interpretability possible, the last layer in the network must have the same number of neurons as there are features in the dataset. In a regular autoencoder architecture, this makes it possible to train the network with a "reconstruction loss" which aims for the minimization of the difference between the original input $x$ and the output $\hat{x} = f(x)$, where $f$ is a neural network architecture. This is not necessarily the case in our framework. We also add a parameter-less "sum layer" which adds the output of the network to its input, the original features. Another way to think about the required architecture under our framework is as a skip-connection in the fashion of ResNets ([He+16]) between the input and the reconstruction layer (see Figure 5.13).

Constraining the architecture in the aforementioned way has the effect

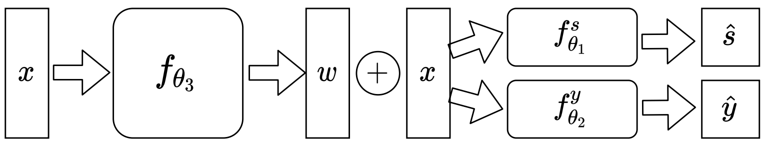


Figure 5.13: A gradient reversal-based neural network constrained for interpretability so to belong in our interpretable framework. The vector $w$ matches in size with $x$, and can then be summed with the original representation $x$ and analyzed for interpretability. This architectural constraint can be applied to other neural architectures.

of making it possible to interpret the neural activations of the last layer in feature space. As mentioned above, our framework is flexible in the sense that many representation learning algorithms can be constrained so to enjoy interpretability properties. To provide a running example, we start from the debiasing models based on the Gradient Reversal Layer of [Gan+16] originally introduced in the domain adaptation context and then employed in fairness by various authors (e.g. [MOW17; Xie+17]). The debiasing effect here is enforced by training a subnetwork $f^{s}_{\theta_1}(z)$ to predict the sensitive attribute and inverting its gradient when backpropagating it through the main network $f(x)$. Another sub-network learns to predict $\hat{y} = f^{y}_{\theta_2}(z)$. Both networks are connected to a main "feature extractor" $z = f_{\theta_3}(x)$. The two models are pitted against one another in extracting useful information for utility purposes (estimating $p(y \mid \hat{x})$) and removing information about $s$. This strategy is similar to the one we employed in fair classification (Section 5.1) and fair ranking (Section 5.2). No modification is needed to the learning algorithm, while the architecture has to be restricted so that the length of the $\hat{x}$ vector is the same as the original features $x$. One concerning factor is whether the neural activations can really be interpreted in feature space, as features can take arbitrary values or be non-continuous (e.g. categorical). We circumvent this issue by coupling the commonly employed feature normalization step and the activation functions of the last neural layer. More specifically, the two functions must map to two coherent intervals of values. As an example, employing standard scaling (feature mean is normalized to 0, standard deviation is normalized to 1) will require an hyperbolic tangent activation function in the last layer. The model will then be enabled in learning a negative or positive correction depending on the sign of the neural activation. It is still possible to use sigmoid activations when the features are normalized in $[0, 1]$ by means of a min-max normalization (lowest value for the feature is 0 and highest is 1). Summing up, the debiasing architecture of Ganin et al. can be modified via the following steps:

1. Normalize the original input features $x_{raw}$ via some normalization function $x = g(x_{raw})$.
2. Set up the neural architecture so that the length of $w = f_{\theta_3}(x)$ is equal to the length of $x$.
3. Add a skip-connection between the input and the reconstruction layer.

After training, the corrected vectors $z = f_{\theta_3}(x) + x$ and the correction vectors $w = f_{\theta_3}(x)$ can be interpreted in feature space by computing the inverse normalization $\hat{x}_{raw} = g^{-1}(z)$ and $w_{raw} = g^{-1}(w)$.

Other neural algorithms can be modified similarly so to belong in the interpretable fair framework, and similar steps can be applied to e.g. the Variational Fair Autoencoder of Louizos et al. [Lou+15] and the variational bound-based objective of Moyer et al. [Moy+18]. In our experiments, we will however focus on the state-of-the-art fair ranking model introduced in the previous Chapter 5.2 [Cer+20], which is based on the gradient reversal layer by Ganin et al. [Gan+16].

### 5.3.1 Experiment and Results

As explained in the previous Section, to constrain our state of the art fair ranker for interpretability it is sufficient to constrain its architecture to extract features which have the same dimensionality as $X$. After adding the skip-connection between the first layer and the last feature extraction layer, no other changes are needed to the training algorithm, which we leave unchanged from the original work. Therefore, we train the model employing SGD and select hyperparameters (the number of hidden layers; the fairness-relevance parameter $\gamma$; the learning rate for SGD) employing a nested cross-validation with $k = 3$. We relied on the Bayesian Optimization implementation provided by Weights & Biases ([Bie20]) and stopped after 200 model fits. To evaluate the models, we computed their nDCG, rND (a disparate impact fairness metric defined by Ke et al. [YS17]) and GPA (a disparate mistreatment metric defined by Narasimhan et al. [Nar+20]) in Table 1. We selected the best models with the assumption that each metric has equal importance.

We focus our discussion on the COMPAS dataset, one of the most popular datasets in fair classification and ranking. We employ, as widely done in the literature, the 10 COMPAS classes as relevance classes for our ranking algorithm. On top of evaluating relevance (via nDCG) and fairness (via rND as introduced by [YS17]), we analyze the correction vectors focusing on the "priors_count". This feature represents the number of previous crimes committed by an individual, and investigating how a fair model changes this feature over the two different groups available (white and black people)

| | | Fair DR | Interpretable Fair DR |
|---|---|---|---|
| COMPAS | 1-rND | 0.841411 ± 0.073 | 0.822243 ± 0.065 |
| | 1-GPA | 0.927383 ± 0.034 | 0.939985 ± 0.036 |
| | nDCG@500 | 0.474789 ± 0.085 | 0.526513 ± 0.067 |
| Bank | 1-rnd | 0.813426 ± 0.004 | 0.812811 ± 0.023 |
| | 1-GPA | 0.918763 ± 0.002 | 0.925992 ± 0.008 |
| | nDCG@500 | 0.671236 ± 0.005 | 0.652672 ± 0.014 |

Table 5.3: Results for the experimentation performed on the COMPAS and Bank datasets. For all the metrics, higher is better. We observe that the performance of the Fair DirectRanker (Fair DR) introduced here in Section 5.2.2 is not impacted meaningfully when constrained for interpretability.

| | Black | White | Avg. Difference |
|---|---|---|---|
| priors_count, original | 2.406494 | 1.578894 | 0.8276 |
| priors_count, corrected | 2.211587 (-8.1%) | 1.486147 (-5.8%) | 0.72544 (-12.4%) |

Table 5.4: Average values for the *priors_count* feature in the COMPAS dataset over the two ethnicity groups. We observe *disparate corrections*, i.e. black individuals receive a stronger negative correction.

can provide insights into the model's reasoning. We provide the average correction value for our best model in Table 5.3.1. In the table, we observe that making the two groups more similar translates into *disparate corrections* which impact black people more than white. Here, one could make the argument that changing the attributes of individuals is equivalent to rewriting their personal history, and could be seen as unlawful. This objection merits attention, and needs to be investigated further. At this time, we would posit that this issue is common to all fair representation learning algorithms, with the difference that the correction is computed by projecting individuals' data into non-readable latent dimensions. The benefit of our framework is that it is possible to investigate this transformation, and possibly refuse the decisions if they are seen as problematic.

## 5.4 Invariant-Shared Feature Spaces with Normalizing Flows

A common thread between all the techniques presented so far is one of *information removal*. That is, the objective of fairness and invariance is pursued by means of gradient reversal. As discussed in the previous sections, this concept, when combined with other information removal techniques, can provide a fairness constraint on the learning of neural network models.

In this Section, I present a contribution which leverages *normalizing flows*, an invertible neural network-like model that can map an arbitrary data distribution to a known one for which density estimates are trivial to compute (e.g., a Gaussian). This methodology has been published as a workshop paper at the International Conference for Learning Representations 2021.

Our fair representation algorithm learns two models, one on all the data available ($G_A$ trained on $\mathbf{x}$) and one on the data coming from a single "pivot" group ($G_P$ trained on $\mathbf{x_p}$); furthermore, the latent feature space $Z$ of the two models is constrained to be the same. By leveraging the invertibility of normalizing flows, our algorithm first transforms all the data into the latent feature space $Z$; then, it uses the inverse transformation $G_P^{-1}(\mathbf{z})$ to obtain *shared* representations. In plain words, any individual belonging to a group $G_i$ may be "translated" into the feature space of pivot group $G_p$. The base concept of this approach is therefore to learn a shared feature space between groups.

### 5.4.1 Normalizing Flows

For transforming data to some latent space, normalizing flows can be used. The learned transformation is defined as $h : Z \to X$, where $X$ is the original data and $Z$ the latent space. One property of this mapping is that it is invertible. As formulated by [Gro+19] one can use the change-of-variables formula to relate $p_X$ and $p_Z$, the marginal densities of $X$ and $Z$, respectively, by:

$$p_X(x) = p_Z(h^{-1}(x)) \left| \det \frac{\partial h^{-1}(x)}{\partial x^T} \right| . \tag{5.11}$$

Recent work has shown that such transformations can be done by deep neural networks, for example, NICE ([DKB15a]) and Autoregressive Flows ([Kin+16]). Furthermore it was shown that evaluating likelihoods via the change-of-variables formula can be done efficiently by employing an architecture based on invertible coupling layers ([DSB17]). The aforementioned architecture is named RealNVP and is the base building block for our methodology.

In domain adaptation, the currently developed AlignFlow by [Gro+19], which is a latent variable generative framework that uses normalizing flows ([RM16; DKB15a; DSB17]), is used to transform samples from one domain to another. The data from each domain was modeled with an invertible generative model having a single latent space over all domains. In the context of domain adaptation, the domain class is also known during testing, which is not the case for fairness. Nevertheless, the general idea of training two Real NVP models is used for AlignFlow as well for the model proposed here.

### 5.4.2 The Fair Normalizing Flow Framework

In this section we describe in detail our algorithm and how it can be employed to obtain fair representations. We define as $X$ the input feature space and the data vectors as $x$. In group fairness, one also has at least one variable $s$ for each data vector, representing a piece of sensitive information such as ethnicity, gender or age. This allows for the definition of a number of groups $G_1 \dots G_{|s|}$, one for each possible value of $s$. Lastly, labels $y$, representing either categorical (classification) or ordinal (ranking) information may be available in a supervised setting, which is usually the case in fair representation learning. The basic building block of our contribution is the normalizing flow model. A normalizing flow model can learn a bijection $f : X \to Z$, where $Z$ is a latent feature space which may be sampled easily, such as an isotropic Gaussian distribution. Then, it is possible to estimate the density $p_X$ via $f$ and the change-of-variables formula:

$$p_X(x) = p_Z(f(x)) \left| \det \frac{\partial f}{\partial x^T} \right| . \tag{5.12}$$

Our approach is to first select a pivot group $G_p$ out of the $|s|$ available ones. Then, a normalizing flow model $f_p$ is trained to learn a bijection $X_{G_p} \longleftrightarrow Z$, where we denote $X_{G_p}$ as the feature space for group $G_p$. Then, another normalizing flow model $f_{all}$ is trained independently to learn a bijection between the feature space $X$ to the same known distribution $Z$: $f_{all} : X \longleftrightarrow Z$. This makes it possible to relate the three densities $p_X$, $p_{X_{G_p}}$ and $p_Z$ to one another via the change-of-variable formula:

$$p_{X_{G_p}}(x_p) = p_Z(f_p(x_p)) \left| \det \frac{\partial f_p}{\partial x_p^T} \right|, \tag{5.13}$$

$$p_X(x) = p_Z(f_{all}(x)) \left| \det \frac{\partial f_{all}}{\partial x^T} \right|. \tag{5.14}$$

Therefore, one may employ the two models to build a bijection chain $X \overset{f_{all}}{\longleftrightarrow} Z \overset{f_p^{-1}}{\longleftrightarrow} X_p$, allowing for the transformation of vectors $x \in X$ into vectors $x_p \in X_{G_p}$. While a bijection cannot remove information about $s$, as mutual information is in general invariant to invertible transformations, the procedure is helpful in practice by obfuscating the difference between $G_p$ and $G_{all}$, as shown in Section 5.4.3.

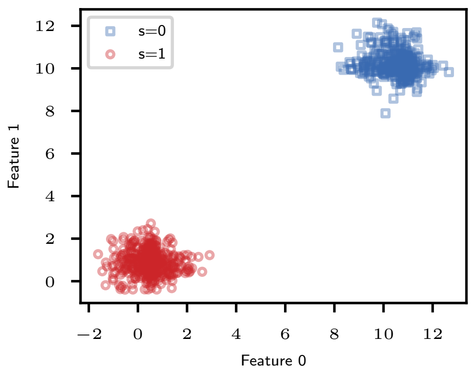
As commonly done in the normalization flow literature, we train our models with a log likelihood loss (see, e.g., [DKB15b] and [PPM17]). We rely on the "coupling layers" architecture, which guarantees invertibility, as introduced by [DSB17].

To ensure that the new representations can still be used to predict the target labels $y$, a classification or ranking model can be trained on top of them. We included a loss evaluated on the target label which is propagated through both normalizing flow models. This leads to a two-fold loss function, which can be regulated by the $\gamma$ hyperparameter to address the relevance-fairness trade-off:

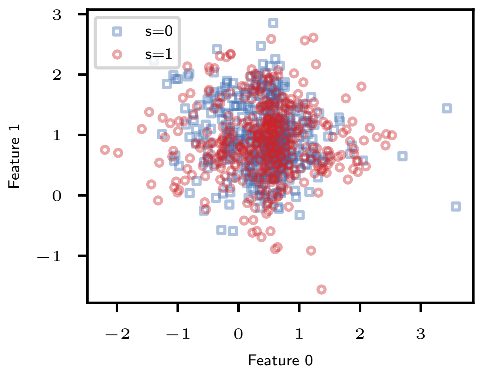
$$L = \gamma(L_{f_p} + L_{f_{all}}) + L_y. \tag{5.15}$$

Here, $L_y$ can be any loss function which can be evaluated on $y$. The gradients of $L_{f_p}$ and $L_{f_{all}}$ are only applied to $f_p$ and $f_{all}$, respectively, while the ones of $L_y$ are applied to $f_p$, $f_{all}$ and the model predicting the target labels $y$.

To give a further intuition about this model, we show its performance on a simple toy dataset in Figure 5.14. In this dataset, we define two different gaussian distributions, each one representing a group. The mean of the two gaussians is very different, and any clustering algorithm would have no issue finding information about the sensitive attribute here. By employing our methodology, it is apparent that the point cloud for $s = 0$ is mapped onto the other. The heavy overlap between the two makes it difficult to distinguish between the groups.



(a) Original toy data



(b) Toy data after applying the framework

Figure 5.14: In 5.14(a) the generated toy data is shown where the different groups are displayed with blue and red. In 5.14(b) the transformed toy data is shown using the FairNF algorithm.

### 5.4.3 Experiments and Results

To overcome statistical fluctuations during the experiments, we split the datasets into 3 internal and 3 external folds. On the 3 internal folds, a

Bayesian grid search, optimizing the fairness measure, is used to find the best hyperparameter setting. The best setting is then evaluated on the 3 external folds.

| Model | Dataset | 1-rND | 1-GPA | NDCG@500 |
|---|---|---|---|---|
| FairNF | Compas | 0.838 ± 0.059 | 0.934 ± 0.030 | 0.474 ± 0.115 |
| FairNF | Adult | 0.922 ± 0.009 | 0.929 ± 0.025 | 0.460 ± 0.054 |
| FairNF | Bank | 0.859 ± 0.017 | 0.892 ± 0.016 | 0.519 ± 0.011 |
| AdvDR | Compas | 0.864 ± 0.061 | 0.911 ± 0.047 | 0.424 ± 0.033 |
| AdvDR | Adult | 0.884 ± 0.036 | 0.975 ± 0.016 | 0.647 ± 0.087 |
| AdvDR | Bank | 0.778 ± 0.106 | 0.735 ± 0.036 | 0.521 ± 0.086 |
| AdvCls | Compas | 0.823 ± 0.057 | 0.917 ± 0.032 | 0.542 ± 0.055 |
| AdvCls | Adult | 0.929 ± 0.007 | 0.901 ± 0.021 | 0.629 ± 0.008 |
| AdvCls | Bank | 0.918 ± 0.051 | 0.972 ± 0.026 | 0.198 ± 0.176 |
| DELTR | Compas | 0.825 ± 0.072 | 0.926 ± 0.007 | 0.438 ± 0.202 |
| DELTR | Adult | 0.744 ± 0.087 | 0.742 ± 0.048 | 0.142 ± 0.119 |
| DELTR | Bank | 0.823 ± 0.057 | 0.917 ± 0.032 | 0.542 ± 0.055 |

Table 5.5: Results for different fairness datasets. The used fairness metrices are 1-rND and 1-GPA, as introduced previously, and the commonly used relevance metric NDCG@k where we set $k = 500$.

We compare our model, which we call FairNF in Table 5.5, with the Fair Adversarial DirectRanker (AdvDR in the rest of this section), the Debiasing Classifier introduced in Section 5.1 (AdvCls) and a fair listwise ranker (DELTR [ZC19]).

Since it is from a theoretical point (to the best of our knowledge) not clear how exactly a Real NVP model is treating non-continuous discrete features, we designed two different experiments whether the model is able to be trained on discrete features. For the first experiment done on the Compas dataset, we included discrete and continuous features. For the second experiment done on the Adult (see [Koh96]) and Bank dataset (see [MCR14a]) we only used continuous features.

For our implementation of the models and the experimental setup, see *https://zenodo.org/record/4566895*.

In Table 5.5 the results for different fairness datasets are shown. FairNF is the proposed algorithm explained in Section 5.4.2. In terms of the two fairness measures, this algorithm is outperforming the other approaches in at least one measure over all datasets. The only algorithm that is better than FairNF in terms of fairness is AdvCls on the Bank dataset. However, on this dataset the algorithm is performing poorly on NDCG@500, while FairNF has a stable performance. In terms of relevance, FairNF and AdvDR have similar performance on the Compas and Bank dataset, while AdvDR is better on the Adult one. The other two algorithms may have weaker results on relevance (AdvCls on Bank; DELTR on Adult). Looking at the question whether the model is able to be trained on continuous features only, we cannot see any significant performance difference comparing the experiments done on Compas compared with the ones done on Adult and Bank.

# Chapter 6

# Conclusions

In this thesis I have explored the concept of invariance in neural network representations and how it relates to fairness. A new building block for architectures which obtain invariant representations has been developed, the noise module (Section 4.2); this module is part of architectures which display state of the art results in fair classification (Section 5.1) and fair ranking (Section 5.2). I have also proposed an adaptation of neural architectures for invariance and fairness so that they are also interpretable (Section 5.3). Lastly, I have presented an invertible architecture which can map data points from different groups into a single one (Section 5.4).

Of course, the challenges in fair invariant learning are still plentiful. Perhaps the clearest line of research here is developing techniques to relate the training objectives of neural networks to different fairness definitions. While in practice the adversarial loss of Ganin et al. [Gan+16] can generalize to different fairness definitions via meta-optimization (i.e. hyperparameter selection), optimizing different definitions directly would be an important direction to pursue. The issue is that metrics for e.g. disparate impact and mistreatment are not differentiable. In fair ranking, a possible workaround would be to borrow the pseudo-gradient concept as introduced by Burges et al. in their LambdaRank model [BRL07]. LambdaRank estimates gradients as the change in a non-differentiable relevance measure after swapping the position of two documents in a query. An extension of this idea to fairness metrics would provide a direct way to optimize for disparate impact or disparate mistreatment depending on the application requirements.

Interpretability in invariant representation learning is also a concept that needs to be explored further. While the idea of learning feature corrections is certainly attractive, the current framework "breaks" integer-valued features as it learns a real-valued correction. As an example, an individual with 3 previous crimes in the COMPAS dataset might be corrected to have committed 2.8 crimes in the past. This has no clear real-world meaning. One line of investigation here is exploring threshold activations as in Rosenblatt's Perceptron. Optimization via gradient descent is challenging, as the step function is not continuous; however, some authors (see Bengio et al. [BLC13], citing a Hinton lecture) propose to treat it as if it were a linear function during the backward pass.

More in general, fairness in neural networks remains a critical field of research. As non-discrimination and transparency in ML models may be required by law, the applicability of unconstrained models might become more limited in practice. While invariance in neural representations is only one possible way to investigate this issue, it may play a central role in the future of neural network models.

# Bibliography

[AB18] A. Adadi and M. Berrada. "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)". In: *IEEE Access* 6 (2018), pp. 52138–52160. ISSN: 2169-3536.

[ACL19] NY Chapter of the ACLU. *Stop-and-Frisk Data.* 2019. URL: `https://www.nyclu.org/en/stop-and-frisk-data`.

[Alm03] Luıs B Almeida. "MISEP–Linear and nonlinear ICA based on mutual information". In: *Journal of Machine Learning Research* 4.Dec (2003), pp. 1297–1318.

[Ang+16] Julia Angwin et al. *Machine Bias.* 2016. URL: `https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing`.

[Bel+18] Mohamed Ishmael Belghazi et al. "Mutual information neural estimation". In: *International Conference on Machine Learning.* PMLR. 2018, pp. 531–540.

[Bie20] Lukas Biewald. *Experiment Tracking with Weights and Biases.* Software available from wandb.com. 2020. URL: `https://www.wandb.com/`.

[BLC13] Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. “Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation”. In: *CoRR* abs/1308.3432 (2013). arXiv: `1308.3432`. URL: `http://arxiv.org/abs/1308.3432`.

[BR18] M. Bogen and A. Rieke. *Help wanted: an examination of hiring algorithms, equity, and bias.* 2018. URL: `https://www.upturn.org/reports/2018/hiring-algorithms/`.

[BRL07] Christopher Burges, Robert Ragno, and Quoc Le. “Learning to Rank with Nonsmooth Cost Functions”. In: *Advances in Neural Information Processing Systems.* Ed. by B. Schölkopf, J. Platt, and T. Hoffman. Vol. 19. MIT Press, 2007. URL: `https://proceedings.neurips.cc/paper/2006/file/af44c4c56f385c43f2529f9b1b0 Paper.pdf`.

[Bur+05] Chris Burges et al. “Learning to Rank Using Gradient Descent”. In: *ICML.* 2005.

[Cao+06] Yunbo Cao et al. “Adapting ranking SVM to document retrieval”. In: *ACM SIGIR.* 2006.

[Cao+07a] Zhe Cao et al. “Learning to Rank: From Pairwise Approach to Listwise Approach”. In: *ICML.* 2007.

[Cao+07b] Zhe Cao et al. “Learning to rank: from pairwise approach to listwise approach”. In: *Proceedings of the 24th international conference on Machine learning.* 2007, pp. 129–136.

[Car97] Rich Caruana. “Multitask learning”. In: *Machine learning* 28.1 (1997), pp. 41–75.

[CEL20] Mattia Cerrato, Roberto Esposito, and Laura Li Puma. “Constraining Deep Representations with a Noise Module for Fair Classification”. In: *ACM SAC*. 2020.

[Cer+20] M. Cerrato et al. “Fair pairwise learning to rank”. In: *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*. 2020, pp. 729–738. DOI: `10.1109/DSAA49011.2020.00083`.

[CGD92] William S Cooper, Fredric C Gey, and Daniel P Dabney. “Probabilistic retrieval based on staged logistic regression”. In: *ACM SIGIR*. 1992.

[CLM20] Lee Cohen, Zachary C. Lipton, and Yishay Mansour. “Efficient Candidate Screening Under Multiple Tests and Implications for Fairness”. In: *1st Symposium on Foundations of Responsible Computing, FORC 2020, June 1-3, 2020, Harvard University, Cambridge, MA, USA (virtual conference)*. Ed. by Aaron Roth. Vol. 156. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020, 1:1–1:20. DOI: `10.4230/LIPIcs.FORC.2020.1`. URL: `https://doi.org/10.4230/LIPIcs.FORC.2020.1`.

[CPC19] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso. “Machine Learning Interpretability: A Survey on Methods and Metrics”. In: *Electronics* 8.8 (2019), p. 832.

[CT06] Thomas M Cover and Joy A Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. 2006.

[Cyb89] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[Den+09] J. Deng et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: *CVPR09*. 2009.

[DG17] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: `http://archive.ics.uci.edu/ml`.

[Dix+18] Lucas Dixon et al. "Measuring and Mitigating Unintended Bias in Text Classification". In: 2018.

[DKB15a] Laurent Dinh, David Krueger, and Yoshua Bengio. *NICE: Non-linear Independent Components Estimation*. 2015. arXiv: `1410.8516 [cs.LG]`.

[DKB15b] Laurent Dinh, David Krueger, and Yoshua Bengio. "NICE: Non-linear Independent Components Estimation". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: `http://arxiv.org/abs/1410.8516`.

[DMB16] William Dieterich, Christina Mendoza, and Tim Brennan. "COMPAS risk scales: Demonstrating accuracy equity and predictive parity". In: *Northpointe Inc.* (2016).

[DSB17] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using Real NVP". In: *International Conference on Learning Representations*. 2017.

[Dwo+12] Cynthia Dwork et al. "Fairness through awareness". In: *Proceedings of the 3rd innovations in theoretical computer science conference*. 2012, pp. 214–226.

[FN96] Batya Friedman and Helen Nissenbaum. "Bias in Computer Systems". In: *ACM TOIS* 14.3 (1996), pp. 330–347. ISSN: 1046-8188.

[Fri00] Jerome H. Friedman. "Greedy Function Approximation: A Gradient Boosting Machine". In: *Annals of Statistics* 29 (2000), pp. 1189–1232.

[Gan+16] Yaroslav Ganin et al. "Domain-adversarial training of neural networks". In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 2096–2030.

[Gol+19] Ziv Goldfeld et al. "Estimating Information Flow in Deep Neural Networks". In: *ICML*. 2019.

[Goo+14] Ian Goodfellow et al. "Generative adversarial nets". In: *Advances in neural information processing systems*. 2014, pp. 2672–2680.

[Goo17] Ian J. Goodfellow. "NIPS 2016 Tutorial: Generative Adversarial Networks". In: *CoRR* abs/1701.00160 (2017). arXiv: `1701.00160`. URL: `http://arxiv.org/abs/1701.00160`.

[Gre+05] Arthur Gretton et al. "Kernel methods for measuring independence". In: *Journal of Machine Learning Research* 6.Dec (2005), pp. 2075–2129.

[Gre+12] Arthur Gretton et al. "A kernel two-sample test". In: *Journal of Machine Learning Research* 13.Mar (2012), pp. 723–773.

[Gro+19] Aditya Grover et al. *AlignFlow: Cycle Consistent Learning from Multiple Domains via Normalizing Flows.* 2019. arXiv: `1905.12892 [cs.LG]`.

[He+16] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition.* 2016, pp. 770–778.

[Hin] Geoffrey E Hinton. "Distributed representations". In: ().

[KA14] Justin B Kinney and Gurinder S Atwal. "Equitability, mutual information, and the maximal information coefficient". In: *Proceedings of the National Academy of Sciences* 111.9 (2014), pp. 3354–3359.

[KC09] Faisal Kamiran and Toon Calders. "Classifying without discriminating". In: *2009 2nd International Conference on Computer, Control and Communication.* IEEE. 2009, pp. 1–6.

[Kin+16] Durk P Kingma et al. "Improved Variational Inference with Inverse Autoregressive Flow". In: *Advances in Neural Information Processing Systems.* Ed. by D. Lee et al. Vol. 29. Curran Associates, Inc., 2016, pp. 4743–4751. URL: `https://proceedings.neurips.cc/paper/2016/file/ddeebdeefdb7e7e7a697e1c3e3d8ef54-Paper.pdf`.

[Koh96] Ron Kohavi. "Scaling Up the Accuracy of Naive-Bayes Classifiers: a". In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining.* 1996.

[Köp+19] Marius Köppel et al. "Pairwise Learning to Rank by Neural Networks Revisited: Reconstruction, Theoretical Analysis and Practical Performance". In: *European Conference on Machine*

*Learning* (2019). arXiv: 1909.02768. URL: http://arxiv.org/abs/1909.02768.

[KSG04] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. "Estimating mutual information". In: *Physical review E* 69.6 (2004), p. 066138.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems.* 2012, pp. 1097–1105.

[KW14] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings.* 2014. arXiv: http://arxiv.org/abs/1312.6114v10 [stat.ML].

[LBH15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521.7553 (2015), p. 436.

[LeC+90] Yann LeCun et al. "Handwritten Digit Recognition with a Back-Propagation Network". In: *Advances in Neural Information Processing Systems.* Ed. by D. Touretzky. Vol. 2. Morgan-Kaufmann, 1990, pp. 396–404. URL: https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf.

[Les+93] Moshe Leshno et al. "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function". In: *Neural networks* 6.6 (1993), pp. 861–867.

[Lev16] Sam Levin. *Artificial Intelligence Beauty Contest Doesn't Like Black People*. 2016. URL: https://www.theguardian.com/technology/2016/sep/08/artificial-intelligence-beauty-contest-doesnt-like-black-people.

[Lin70] Seppo Linnainmaa. *Algoritmin Kumulativinen Pyöristysvirhe Yksittäisten Pyöristysvirheiden Taylor-Kehitelmänä*. Master's Thesis. 1970. URL: http://people.idsia.ch/~juergen/linnainmaa1970thesis.pdf.

[Lin76] Seppo Linnainmaa. "Taylor expansion of the accumulated rounding error". In: *BIT Numerical Mathematics* 16.2 (1976), pp. 146–160.

[Lou+15] Christos Louizos et al. "The Variational Fair Autoencoder". In: *ICLR 2016* abs/1511.00830 (2015).

[LWB08] Ping Li, Qiang Wu, and Christopher J Burges. "Mcrank: Learning to rank using multiple classification and gradient boosting". In: *NIPS*. 2008.

[LXL16] Yinan Li, Ruoyi Xu, and Fang Liu. "Whiteout: Gaussian adaptive regularization noise in deep neural networks". In: *stat* 1050 (2016), p. 5.

[Mal20] Gianclaudio Malgieri. "The Concept of Fairness in the GDPR: A Linguistic and Contextual Interpretation". In: *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. FAT* '20. Barcelona, Spain: Association for Computing Machinery, 2020, pp. 154–166. ISBN: 9781450369367. DOI: 10.1145/3351095.3372868. URL: https://doi.org/10.1145/3351095.3372868.

[MCR14a] Sérgio Moro, Paulo Cortez, and Paulo Rita. “A Data-Driven Approach to Predict the Success of Bank Telemarketing”. In: *Decision Support Systems* 62 (June 2014). DOI: `10.1016/j.dss.2014.03.001`.

[MCR14b] Sérgio Moro, Paulo Cortez, and Paulo Rita. “A data-driven approach to predict the success of bank telemarketing”. In: *Decision Support Systems* 62 (2014), pp. 22–31. ISSN: 0167-9236. DOI: `https://doi.org/10.1016/j.dss.2014.03.001`. URL: `https://www.sciencedirect.com/science/article/pii/S016792361400061X`.

[Meh+19] Ninareh Mehrabi et al. “A Survey on Bias and Fairness in Machine Learning”. In: *arXiv e-prints*, arXiv:1908.09635 (Aug. 2019), arXiv:1908.09635. arXiv: `1908.09635 [cs.LG]`.

[Mik+13] Tomas Mikolov et al. “Efficient Estimation of Word Representations in Vector Space”. In: *preprint arXiv:1301.3781* (2013).

[MOW17] Daniel McNamara, Cheng Soon Ong, and Robert C Williamson. “Provably fair representations”. In: *preprint arXiv:1710.04394* (2017).

[Moy+18] Daniel Moyer et al. “Invariant representations without adversarial training”. In: *Advances in Neural Information Processing Systems* 31 (2018), pp. 9084–9093.

[MP69] Marvin Minsky and Seymour Papert. *Perceptrons.* MIT Press, 1969.

[MP90] WS McCulloch and W Pitts. “A logical calculus of the ideas immanent in nervous activity. 1943.” In: *Bulletin of Mathematical Biology* 52.1-2 (1990), pp. 99–115.

[MRL95] Young-Il Moon, Balaji Rajagopalan, and Upmanu Lall. "Estimation of mutual information using kernel density estimators". In: *Physical Review E* 52.3 (1995), p. 2318.

[Mur12] Kevin P Murphy. *Machine learning: a probabilistic perspective.* MIT press, 2012.

[Nar+20] Harikrishna Narasimhan et al. "Pairwise Fairness for Ranking and Regression". In: *AAAI*. 2020.

[Nit19] Samir Hans Nitin Mittal David Kuder. "AI-fueled organizations: Reaching AI's full potential in the enterprise". In: *Deloitte Insights* (Jan. 2019). URL: `https://www2.deloitte.com/insights/us/en/focus/tech-trends/2019/driving-ai-potential-organizations.html`.

[Oda19] Stephen Odaibo. "Tutorial: Deriving the Standard Variational Autoencoder (VAE) Loss Function". In: *arXiv preprint arXiv:1907.08956* (2019).

[Ols11] Parmy Olson. *The Algorithm That Beats Your Bank Manager.* 2011. URL: `https://www.forbes.com/sites/parmyolson/2011/03/15/the-algorithm-that-beats-your-bank-manager/`.

[Pan03] Liam Paninski. "Estimation of entropy and mutual information". In: *Neural computation* 15.6 (2003), pp. 1191–1253.

[PPM17] George Papamakarios, Theo Pavlakou, and Iain Murray. "Masked Autoregressive Flow for Density Estimation". In: *Advances in Neural Information Processing Systems.* Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017, pp. 2338–2347. URL:

https://proceedings.neurips.cc/paper/2017/file/6c1da886822c67822bcf3679d04369fa-Paper.pdf.

[Res+11] David N Reshef et al. “Detecting novel associations in large data sets”. In: *Science* 334.6062 (2011), pp. 1518–1524.

[RHW86] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. “Learning representations by back-propagating errors”. In: *nature* 323.6088 (1986), pp. 533–536.

[Rig+08] Leonardo Rigutini et al. “A neural network approach for learning object ranking”. In: *International Conference on Artificial Neural Networks*. Springer. 2008, pp. 899–908.

[RM16] Danilo Jimenez Rezende and Shakir Mohamed. *Variational Inference with Normalizing Flows*. 2016. arXiv: 1505.05770 [stat.ML].

[RMW14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. “Stochastic Backpropagation and Approximate Inference in Deep Generative Models”. In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 1278–1286. URL: http://proceedings.mlr.press/v32/rezende14.html.

[Ros58] Frank Rosenblatt. “The perceptron: a probabilistic model for information storage and organization in the brain.” In: *Psychological review* 65.6 (1958), p. 386.

[Sam59] Arthur L Samuel. “Some studies in machine learning using the game of checkers”. In: *IBM Journal of research and development* 3.3 (1959), pp. 210–229.

[Sax+19] Andrew M Saxe et al. "On the information bottleneck theory of deep learning". In: *Journal of Statistical Mechanics: Theory and Experiment* 2019.12 (2019), p. 124020.

[Sha+14] Ali Sharif Razavian et al. "CNN features off-the-shelf: an astounding baseline for recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops.* 2014, pp. 806–813.

[Sha48] Claude E Shannon. "A mathematical theory of communication". In: *The Bell system technical journal* 27.3 (1948), pp. 379–423.

[Sri+14] Nitish Srivastava et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. URL: `http://jmlr.org/papers/v15/srivastava14a.html`.

[ST14] Noah Simon and Robert Tibshirani. *Comment on "Detecting Novel Associations In Large Data Sets" by Reshef Et Al, Science Dec 16, 2011.* 2014. arXiv: `1401.7645 [stat.ME]`.

[Str+98] Steven P Strong et al. "Entropy and information in neural spike trains". In: *Physical review letters* 80.1 (1998), p. 197.

[Tan+18] Chuanqi Tan et al. "A survey on deep transfer learning". In: *International conference on artificial neural networks.* Springer. 2018, pp. 270–279.

[TZ15] Naftali Tishby and Noga Zaslavsky. "Deep learning and the information bottleneck principle". In: *2015 IEEE Information Theory Workshop (ITW).* IEEE. 2015, pp. 1–5.

[Vla19] Kelsey Vlamis. *What are beauty pageants really like for black women?* 2019. URL: `https://www.bbc.com/news/world-us-canada-50743904`.

[VR18] Sahil Verma and Julia Rubin. "Fairness definitions explained". In: *2018 ieee/acm international workshop on software fairness (fairware)*. IEEE. 2018, pp. 1–7.

[Wer82] Paul J. Werbos. "Applications of advances in nonlinear sensitivity analysis". In: *System Modeling and Optimization*. Ed. by R. F. Drenick and F. Kozin. Berlin, Heidelberg: Springer Berlin Heidelberg, 1982, pp. 762–770. ISBN: 978-3-540-39459-4.

[WRC98] L.F. Wightman, H. Ramsey, and Law School Admission Council. *LSAC national longitudinal bar passage study*. LSAC research report series. Law School Admission Council, 1998.

[Wu+10] Qiang Wu et al. "Adapting boosting for Information Retrieval Measures". In: *Information Retrieval* 13 (2010), pp. 254–270.

[WW05] Doris Weichselbaumer and Rudolf Winter-Ebmer. "A meta-analysis of the international gender wage gap". In: *Journal of Economic Surveys* 19.3 (2005), pp. 479–511.

[Xie+17] Qizhe Xie et al. "Controllable Invariance through Adversarial Feature Learning". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: `https://proceedings.neurips.cc/paper/2017/file/8cb22bdd0b7ba1ab13d742e22eed8da2-Paper.pdf`.

[XL07] Jun Xu and Hang Li. "AdaRank: A Boosting Algorithm for Information Retrieval". In: *ACM SIGIR*. 2007.

[Yan+18] Ke Yang et al. "A nutritional label for rankings". In: *Proceedings of the 2018 international conference on management of data.* 2018, pp. 1773–1776.

[YS17] Ke Yang and Julia Stoyanovich. "Measuring Fairness in Ranked Outputs". In: *SSDBM '17.* Chicago, IL, USA: Association for Computing Machinery, 2017. ISBN: 9781450352826.

[Yu+20] Shujian Yu et al. "Understanding convolutional neural networks with information theory: An initial exploration". In: *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[Zaf+17] Muhammad Bilal Zafar et al. "Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment". In: *Proceedings of the 26th International Conference on World Wide Web.* International World Wide Web Conferences Steering Committee. 2017, pp. 1171–1180.

[ZC19] Meike Zehlike and Carlos Castillo. "Reducing disparate exposure in ranking: A learning to rank approach". In: *WWW* (2019).

[Zem+13] Rich Zemel et al. "Learning fair representations". In: *International Conference on Machine Learning.* 2013, pp. 325–333.

[Zha+19] Han Zhao et al. "On learning invariant representations for domain adaptation". In: *International Conference on Machine Learning.* PMLR. 2019, pp. 7523–7532.