Neurocomputing 669 (2026) 132498

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A differentiable and uncertainty-aware mutual information regularizer for bias mitigation

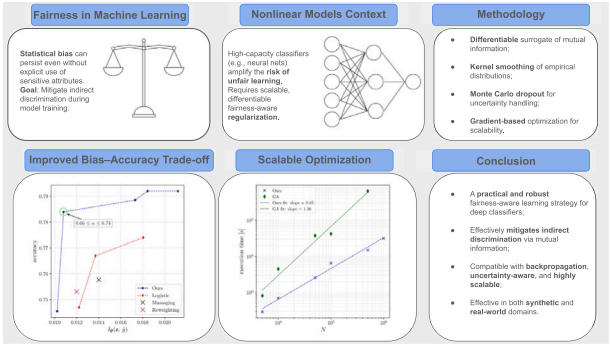
Alessandro Incremona [a], Andrea Pozzi [a,*], Alessandro Guiscardi [a], Daniele Tessera [b]

[a] Faculty of Mathematical, Physical and Natural Sciences, Università Cattolica del Sacro Cuore, Via della Garzetta 48, Brescia, 25133, Italy
[b] Department of Electrical, Computer and Biomedical Engineering, Università di Pavia, Via Adolfo Ferrata 5, Pavia, 27100, Italy

HIGHLIGHTS

- Introduce a differentiable mutual information surrogate for fairness in machine learning classification.
- Combine kernel smoothing and Monte Carlo dropout for robust bias estimation.
- Achieve strong fairness–accuracy trade-offs on synthetic and real datasets.
- Outperform gradient-free methods with sublinear training runtime scaling.

GRAPHICAL ABSTRACT



ARTICLE INFO

Communicated by X. Luo

Keywords:
Regularization
Optimization
Mutual information
Bias mitigation
Neural networks

ABSTRACT

Ensuring algorithmic fairness is a central challenge in high-stakes machine learning applications, where biased predictions can have harmful societal consequences. Discrimination may persist even when sensitive features are excluded from model inputs, due to statistical dependencies between protected and ostensibly non-sensitive attributes that allow bias to be indirectly learned. Mutual information is a principled measure of such dependence, yet its non-differentiability under hard-threshold classification rules precludes direct use in gradient-based training. This paper proposes a fairness-aware learning strategy that embeds a differentiable surrogate of mutual information directly into the objective of nonlinear classifiers. The surrogate replaces hard label assignment with a Bernoulli relaxation of predicted probabilities, stabilizes empirical distributions via kernel smoothing, and accounts for epistemic uncertainty through Monte Carlo dropout, yielding an uncertainty-aware regularizer compatible with stochastic gradient optimization. Experiments on synthetic and real-world datasets show that the method achieves favorable fairness–accuracy trade-offs, with substantial bias reduction and minimal impact on predictive performance. Additional evaluations confirm robustness under counterfactual perturbations and reveal sublinear runtime growth with dataset size, contrasting favorably with the superlinear scaling of a genetic algorithm baseline. Overall, the framework offers a general, theoretically grounded, and practically viable approach to fairness-aware learning in complex, high-dimensional settings, supporting responsible algorithmic decision-making alongside expressive modeling.

* Corresponding author.
Email address: andrea.pozzi@unicatt.it (A. Pozzi).

https://doi.org/10.1016/j.neucom.2025.132498
Received 3 June 2025; Received in revised form 21 November 2025; Accepted 19 December 2025
Available online 22 December 2025

1. Introduction

Fairness in machine learning has emerged as a critical area of research at the intersection of ethics, data science, and algorithmic design. As predictive models are increasingly deployed in socially sensitive contexts—such as hiring, lending, law enforcement, and healthcare—their potential to reinforce or even exacerbate systemic biases has drawn growing scrutiny [1,2]. These concerns are not merely philosophical; unfair decision-making can result in tangible harms, including discrimination against marginalized populations and a loss of public trust in automated systems. High-profile cases, such as racial bias in recidivism risk assessments, have highlighted the societal damage that can arise when models perpetuate historical inequities, emphasizing fairness as a fundamental requirement for the responsible deployment of artificial intelligence [3,4].

Beyond its ethical and societal implications, achieving fairness in machine learning presents significant technical challenges. These difficulties have motivated efforts across research, policy, and industry to develop robust and scalable mitigation strategies [5]. One of the primary obstacles lies in balancing predictive performance with fairness guarantees: models optimized solely for accuracy may learn to exploit spurious correlations that mirror societal disparities, while models constrained to meet fairness criteria may experience reduced utility, particularly when base rates differ across groups [6].

Fairness itself is not a monolithic concept. Multiple formal definitions have been proposed, generally falling into two broad categories: group-level and individual-level fairness. Group fairness notions, such as demographic parity and equalized odds, require statistical parity in model outcomes across predefined protected groups. In contrast, individual fairness aims to ensure that similar individuals are treated similarly, often requiring more granular similarity metrics and careful feature representation [7,8]. These definitions can be mutually incompatible, and choosing among them requires contextual sensitivity and normative judgment. Moreover, enforcing a chosen criterion within the learning pipeline is nontrivial. Simply removing sensitive attributes rarely suffices, as ostensibly non-sensitive features may exhibit strong correlations with protected variables and act as proxies, thereby enabling indirect discrimination [9,10].

These challenges are further compounded by the increasing complexity and opacity of modern machine learning models, particularly deep neural networks. Although such models learn expressive representations, their "black-box" nature hinders interpretability and complicates efforts to verify whether predictions are distributed fairly across demographic groups.

To address these concerns, a range of bias mitigation techniques has been developed, commonly grouped into pre-processing, in-processing, and post-processing methods. Pre-processing approaches alter the training data—for example, via reweighting or massaging [11]—but may obscure underlying structure and often lack adaptability to specific architectures. Post-processing methods modify decision thresholds or outputs with minimal changes to internal mechanics [12], yet can compromise transparency and calibration and may yield inconsistent outcomes for similar individuals. In contrast, in-processing techniques integrate fairness constraints directly into training, introducing fairness-aware objectives or regularization terms that seek to balance accuracy and fairness from the outset [13].

Despite their promise, in-processing methods face their own set of challenges. A central difficulty lies in the choice of an appropriate metric to quantify bias. Mutual information (MI) has gained traction in fairness-aware learning due to its ability to capture general statistical dependencies [14,15], including nonlinear relationships that traditional correlation measures may fail to detect [16]. However, using MI as a regularizer introduces two intertwined difficulties: the non-differentiability of discrete prediction rules and the instability of empirical MI estimates in high-dimensional or sparse settings.

First, MI is often computed using hard predictions or thresholding, which are not compatible with gradient-based optimization. While smoothing techniques or gradient-free optimization methods have been explored [17,18], they can incur substantial computational costs or unstable convergence in nonlinear models.

Second, estimating MI from data requires reliable approximations of joint and marginal distributions, which become noisy in high-dimensional spaces typical of neural networks. Stochastic training procedures further introduce variability, leading to fairness metrics that may fluctuate across training runs [19]. Robustness to model uncertainty is therefore essential for preventing misleading or brittle fairness assessments [20].

In response to these challenges, this paper proposes a novel in-processing bias mitigation method, suitable for application within neural network models, that emphasizes both differentiability and uncertainty-awareness by introducing a regularization term based on MI.

Specifically, the main contributions are the following:

- **Differentiable fairness surrogate:** Introduction of a differentiable, uncertainty-aware surrogate for MI, suitable for use as a regularization term in gradient-based in-processing bias mitigation. The proposed surrogate incorporates Bayesian approximation techniques and kernel smoothing to address data noise in high-dimensional settings and model epistemic uncertainty. The formulation is accompanied by theoretical guarantees on the tightness of the approximation as a lower bound.
- **Integration into nonlinear models:** Integration of the proposed surrogate into the training of nonlinear binary classifiers, demonstrating its adaptability to modern architectures, its ability to achieve favorable fairness-accuracy trade-offs relative to traditional bias mitigation methods, and improved scalability over gradient-free alternatives.
- **Comprehensive evaluation:** A comprehensive experimental evaluation, including both a controlled synthetic dataset and a real-world dataset, to empirically validate the proposed method. In addition, a counterfactual fairness analysis and a sensitivity analysis are conducted in the synthetic setting to further substantiate the robustness of the approach.

The remainder of this paper is organized as follows. Section 2 reviews related work on fairness-aware learning, with a focus on in-processing methods and MI approximation strategies. Section 3 presents the problem formulation and highlights the limitations of classical MI for fairness regularization. Section 4 introduces the proposed methodology, including the construction of the differentiable surrogate and the techniques used for estimation and uncertainty handling. Section 5 describes the experimental setup and presents the results. Section 6 discusses current limitations and potential directions for future work. Finally, Section 7 summarizes the main findings of the paper.

2. Related work

In-processing bias mitigation

In-processing bias mitigation has become a central focus in the algorithmic fairness literature, with numerous strategies proposed to imbue the training process with fairness criteria. A seminal example is the prejudice remover regularizer [13], which adds a penalty term to the classifier's objective to discourage dependence between the predicted outcomes and the sensitive attribute. Although conceptually simple, this and related early approaches often targeted linear or low-capacity models and were tied to specific fairness criteria, limiting their applicability to modern neural networks.

Subsequent work introduced fairness as a constrained optimization problem. For example, Zafar et al. [21] proposed convex proxy constraints for enforcing fairness metrics like disparate impact and disparate

mistreatment in linear classifiers. Similarly, Cotter et al. [22] developed a general framework for optimizing models under rate constraints using proxy-Lagrangian formulations. Beutel et al. [23] introduced an absolute correlation regularizer for deep neural networks that effectively zeroes out linear dependency between a protected attribute and the logits, while [24] employed the Wasserstein-1 distance between outcome distributions for different groups as a differentiable fairness loss. These explicit regularization methods are appealing for their simplicity and flexibility but they can be limited by the fidelity of the chosen surrogate metric. For example, a correlation penalty addresses only linear dependence and might miss nonlinear discrimination, whereas a Wasserstein distance focuses on distributional overlap but might be less sensitive to other statistical relations.

Another influential line employs adversarial training, wherein an auxiliary adversary attempts to infer sensitive attributes from learned representations. The main model is penalized when the adversary succeeds, thus promoting invariance to protected information [25]. While effective, adversarial training introduces instability due to its minimax nature and is sensitive to hyperparameter selection [26].

An alternative minimizes statistical dependence between model outputs and sensitive attributes via kernel-based measures such as the Hilbert-Schmidt Independence Criterion (HSIC) or Maximum Mean Discrepancy (MMD) [27,28]. These regularizers maintain differentiability and avoid adversarial components but can incur high computational costs and kernel sensitivity, especially in large-scale or high-dimensional settings.

Recent extensions include meta-learning for fairness generalization [29], distributionally robust optimization [30], and fairness-aware disentangled representations [31]. Despite progress, in-processing methods continue to face challenges related to scalability, gradient stability, and out-of-domain generalization.

Mutual information approximation

Estimating MI in high-dimensional or nonlinear contexts remains a difficult task. Classical estimators based on histograms or nearest neighbors are non-differentiable and sample-inefficient, making them unsuitable for training deep models [32].

Differentiable surrogates based on variational bounds have therefore been proposed. One widely known approach is MINE [33], which estimates MI by optimizing a neural critic to approximate the density ratio between joint and marginal distributions. Although differentiable, these bounds often suffer from high variance and require careful tuning. CLUB and related methods instead approximate upper bounds using variational marginals, improving stability at the cost of a constant bias [34].

Contrastive learning-based surrogates, such as InfoNCE, estimate MI via auxiliary classification tasks that distinguish true pairs from negative samples [35]. These approximations are more stable in high dimensions but tend to underestimate MI due to loose lower bounds and rely on negative-sampling strategies and batch composition.

Another popular method, Nguyen-Wainwright-Jordan (NWJ), is a variational lower bound that optimizes a function class over joint and product-of-marginals samples [36]. It is theoretically grounded and easier to implement than some other variational bounds, but its performance is highly dependent on the capacity of the function approximator and may degrade under limited data or high noise.

Donsker-Varadhan (DV) bounds represent another classical family of variational estimators based on a dual representation of Kullback-Leibler (KL) divergence [37]. Despite their theoretical elegance, DV-based methods are often unstable in practice due to unbounded objectives and gradient explosions during training.

Reparametrization-based estimators, such as those derived from Barber-Agakov bounds, decompose MI into conditional entropy terms that can be approximated using reparameterizable variational distributions. These methods tend to offer better sample efficiency and tractability, especially when latent variables are involved [38]. However, they require access to tractable posteriors and may suffer from amortization gaps when neural inference networks are used. Some methods approximate MI using mutual predictability frameworks [39]. In such frameworks, the goal is to predict one variable from another using auxiliary models, and the predictive loss serves as a proxy for MI. While intuitive and simple to implement, these methods are fundamentally limited by the expressiveness of the predictive model and may conflate statistical dependence with functional relationship strength, potentially overestimating MI when spurious but learnable patterns exist.

Finally, sliced or projected MI estimators offer scalable alternatives by computing MI over random linear projections of the data [40]. These methods approximate the full MI by aggregating results over several low-dimensional slices. While efficient and differentiable, their accuracy depends on the number and quality of projections used, and they may miss nuanced dependencies not captured by the chosen slices.

The proposed method contributes to this landscape by introducing a differentiable, uncertainty-aware MI surrogate tailored for in-processing bias mitigation. It is designed to be compatible with gradient-based optimization, robust to model uncertainty and data sparsity, and scalable to modern nonlinear architectures.

3. Problem setup

Consider a supervised binary classification setting in which the dataset is defined as

$$D = \{(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}, y^{(i)})\}_{i=1}^{N}, \tag{1}$$

consisting of N samples drawn from an unknown joint distribution $p(\mathbf{x}, \mathbf{z}, y)$. In this formulation, $\mathbf{x} \in \mathcal{X}$ denotes the vector of non-sensitive (i.e., non-protected) features, $\mathbf{z} \in \mathcal{Z}$ represents the sensitive (protected) attributes, and $y \in \mathcal{Y}$ is the target variable. The feature spaces $\mathcal{X}$, $\mathcal{Z}$, and $\mathcal{Y}$ are assumed to be discrete, with the target domain specified as

$$\mathcal{Y} = \{0, 1\}. \tag{2}$$

The conditional distribution $p(y \mid \mathbf{x})$ is assumed to exhibit complex, potentially nonlinear interactions among the components of $\mathbf{x}$, accommodating a wide range of functional dependencies.

The goal is to train a fair, parametric, probabilistic model $p_\theta(y \mid \mathbf{x})$ with parameters $\theta \in \mathbb{R}^d$, to estimate the conditional distribution of the target based solely on non-sensitive features. The probabilistic output of the model is subsequently transformed into a binary classification decision $\hat{y}$ using a thresholding mechanism:

$$\hat{y} = \begin{cases} 1, & \text{if} \quad p_\theta(y = 1 \mid \mathbf{x}) \geq \tau, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Here, τ is the classification threshold, typically set to 0.5, though alternative values may be adopted based on application-specific requirements.

Although the model does not utilize $\mathbf{z}$ as an explicit input, the exclusion of sensitive attributes does not inherently ensure fairness. This is due to potential statistical dependencies between $\mathbf{x}$ and $\mathbf{z}$, which can result in the learned predictions $\hat{y}$ being indirectly influenced by protected information. Such indirect influence may lead to discriminatory outcomes, even when sensitive features are not explicitly provided to the model.

To address this concern, the learning framework jointly optimizes two competing objectives—predictive accuracy and algorithmic fairness— unified within a composite loss function:

$$L(\theta) = (1 - \alpha)\, L_{\mathrm{u}}(\theta) + \alpha\, L_{\mathrm{f}}(\theta), \tag{4}$$

where $L_{\mathrm{u}}(\theta)$ denotes the utility loss, measuring prediction error, and $L_{\mathrm{f}}(\theta)$ represents the fairness loss, introduced to penalize biased behavior in the model's predictions. The scalar hyperparameter $\alpha \in [0, 1]$ governs

the trade-off between these two objectives: lower values of α prioritize predictive performance, while higher values assign greater importance to fairness considerations.

In the context of binary classification, the utility loss $L_{\mathrm{u}}(\theta)$ is typically instantiated as the binary cross-entropy (BCE) loss:

$$L_{\mathrm{u}}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\Big[y^{(i)}\log\big(p_\theta(y=1\mid \mathbf{x}^{(i)})\big) + \big(1-y^{(i)}\big)\log\big(1-p_\theta(y=1\mid \mathbf{x}^{(i)})\big)\Big]. \tag{5}$$

To quantify and mitigate bias, the fairness loss $L_{\mathrm{f}}(\theta)$ is formulated using a dependency measure that captures the statistical association between the model's predicted label $\hat{y}$ and the sensitive attributes $\mathbf{z}$. Mutual information (MI) is employed as the foundational metric due to its capacity to measure general statistical dependence without assuming specific forms of interaction.

Formally, the MI between two discrete random variables A and B is defined as

$$I(A,B) = \sum_{a\in\mathcal{A}}\sum_{b\in\mathcal{B}} p(a,b)\log\left(\frac{p(a,b)}{p(a)\,p(b)}\right), \tag{6}$$

where $\mathcal{A}$ and $\mathcal{B}$ denote the support sets of A and B, respectively. This expression can be equivalently rewritten as the KL divergence between the joint distribution $p(a,b)$ and the product of its marginals $p(a)\,p(b)$:

$$I(A,B) := D_{\mathrm{KL}}\big(p(a,b)\,\|\,p(a)\,p(b)\big), \tag{7}$$

highlighting that MI measures the deviation of the joint distribution from statistical independence. A value of zero indicates independence between A and B, whereas higher values reveal stronger statistical associations.

In the context of fairness, MI is used to quantify the dependence between the predicted label and the sensitive attributes. Letting $A \equiv \mathbf{z}$ and $B \equiv \hat{y}$, the MI becomes

$$I_\theta(\mathbf{z},\hat{y}) = \sum_{\mathbf{z}\in\mathcal{Z}}\sum_{\hat{y}\in\{0,1\}} p_\theta(\mathbf{z},\hat{y})\log\left(\frac{p_\theta(\mathbf{z},\hat{y})}{p(\mathbf{z})\,p_\theta(\hat{y})}\right), \tag{8}$$

where the subscript θ denotes that the joint and marginal distributions are influenced by the model parameters through the predicted outcome $\hat{y}$. A high value of $I_\theta(\mathbf{z},\hat{y})$ suggests that the classifier's predictions are statistically dependent on sensitive attributes, potentially indicating bias. Conversely, a value of zero implies independence and aligns with fairness desiderata.

When the sensitive attribute $\mathbf{z}$ is multidimensional, a weighted sum of marginal MI terms can be adopted to enable component-wise fairness control:

$$I_\theta^{\mathrm{cw}}(\mathbf{z},\hat{y}) := \sum_{j=1}^{n_{\mathbf{z}}}\beta_j\, I_\theta(z_j,\hat{y}), \tag{9}$$

where $n_{\mathbf{z}}$ is the number of sensitive features, z_j denotes the j-th component, and β_j is a non-negative weighting factor. This component-wise adaptation is not, in general, equal to the joint MI $I_\theta(\mathbf{z},\hat{y})$ unless the components z_j are mutually independent; nonetheless, it provides a practical mechanism for enforcing fairness across individual sensitive attributes. In the present framework, the joint MI is used to quantify bias, while the component-wise variant is noted as a possible extension for applications requiring per-attribute control.

While MI is a powerful tool for detecting statistical dependence and, by extension, bias, its direct use in the composite objective function (4) is hindered by non-differentiability. In particular, the hard thresholding in (3) introduces discontinuities that prevent gradient-based optimization.

To overcome these obstacles, a differentiable surrogate for MI is introduced. This surrogate retains the interpretability and effectiveness of the original metric while enabling seamless integration into gradient-based training routines. Furthermore, the formulation is designed to be robust under uncertainty and to perform reliably in scenarios involving noisy, high-dimensional, or sparse input data.

4. Methodology

The proposed methodology builds upon core principles from statistical dependence and information theory, introducing a differentiable variant of MI tailored for optimization with respect to model parameters. This reformulation permits the direct use of MI as a fairness objective within gradient-based learning algorithms. To enhance robustness in practical settings—particularly in the presence of noisy data and model uncertainty—the framework incorporates kernel-based and Bayesian techniques.

4.1. Differentiable relaxation

The derivation begins by revisiting the MI expression in (8), with emphasis on components dependent on the model parameters. In particular, the joint distribution term $p_\theta(\mathbf{z},\hat{y})$ can be factorized using the chain rule as:

$$p_\theta(\mathbf{z},\hat{y}) = p_\theta(\hat{y}\mid\mathbf{z})\,p(\mathbf{z}). \tag{10}$$

Since the model only receives non-sensitive features $\mathbf{x}$ as input, the conditional independence relation

$$p_\theta(\hat{y}\mid\mathbf{x},\mathbf{z}) = p_\theta(\hat{y}\mid\mathbf{x}), \qquad \hat{y}\perp\!\!\!\perp\mathbf{z}\mid\mathbf{x} \tag{11}$$

holds by construction. Using the definition of conditional probability, one obtains:

$$p_\theta(\hat{y},\mathbf{x}\mid\mathbf{z}) = p_\theta(\hat{y}\mid\mathbf{x},\mathbf{z})\,p(\mathbf{x}\mid\mathbf{z}) \tag{12}$$

and by applying the law of total probability, the marginal conditional distribution becomes:

$$p_\theta(\hat{y}\mid\mathbf{z}) = \sum_{\mathbf{x}\in\mathcal{X}} p_\theta(\hat{y},\mathbf{x}\mid\mathbf{z}). \tag{13}$$

Substituting (12) into (13) and applying the independence relation (11) leads to the expression:

$$p_\theta(\hat{y}\mid\mathbf{z}) = \sum_{\mathbf{x}\in\mathcal{X}} p_\theta(\hat{y}\mid\mathbf{x})\,p(\mathbf{x}\mid\mathbf{z}). \tag{14}$$

Despite the dependence of $p_\theta(\hat{y}\mid\mathbf{x})$ on the model parameters, the resulting function is not differentiable. Specifically, due to the deterministic thresholding rule defined in (3), the predicted label $\hat{y}$ is obtained as

$$g_\theta(\mathbf{x}) = \mathbf{1}\big\{\pi_\theta(\mathbf{x}) \geq \tau\big\}, \tag{15}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function and $\pi_\theta(\mathbf{x})$ denotes the predicted probability:

$$\pi_\theta(\mathbf{x}) := p_\theta\big(y=1\mid\mathbf{x}\big). \tag{16}$$

Under this definition, the conditional distribution of $\hat{y}$ given $\mathbf{x}$ becomes a Kronecker delta:

$$p_\theta(\hat{y}\mid\mathbf{x}) = \delta_{\hat{y},\,g_\theta(\mathbf{x})} = \begin{cases} 1, & \text{if}\quad \hat{y} = g_\theta(\mathbf{x}),\\ 0, & \text{otherwise.}\end{cases} \tag{17}$$

As a function of the model parameters θ, this distribution is piecewise constant, changing only at the decision boundary defined by the set $\big\{\theta : \pi_\theta(\mathbf{x}) = \tau\big\}$. Consequently, its gradient is either zero or undefined:

$$\nabla_\theta p_\theta(\hat{y}\mid\mathbf{x}) = \begin{cases} \text{undefined}, & \text{if}\quad \pi_\theta(\mathbf{x}) = \tau,\\ \mathbf{0}, & \text{otherwise.}\end{cases} \tag{18}$$

This vanishing or undefined gradient propagates to any quantity that depends on $p_\theta(\hat{y}\mid\mathbf{x})$, including the MI term, thus rendering direct optimization via backpropagation infeasible.

To enable gradient-based optimization, a continuous relaxation of the hard predictor is introduced via Bernoulli modeling of the predicted label. Recall the class-probability score defined in (16), which appears in the deterministic thresholding rule in (15). Rather than assigning a label deterministically, the relaxed model treats the output as a random variable:

$$\hat{y} \mid \mathbf{x} \sim \text{Bernoulli}\big(\pi_\theta(\mathbf{x})\big), \tag{19}$$

which yields the following conditional likelihood:

$$\hat{p}_\theta(\hat{y} \mid \mathbf{x}) = \pi_\theta(\mathbf{x})^{\hat{y}} \left[1 - \pi_\theta(\mathbf{x})\right]^{1-\hat{y}}. \tag{20}$$

Notably, the resulting expression is continuously differentiable with respect to θ.

Replacing the original non-differentiable term $p_\theta(\hat{y} \mid \mathbf{x})$ with its relaxed counterpart $\hat{p}_\theta(\hat{y} \mid \mathbf{x})$ in the law of total probability formulation (14) leads to a soft, parameter-dependent conditional distribution:

$$\hat{p}_\theta(\hat{y} \mid \mathbf{z}) = \sum_{\mathbf{x} \in \mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, p(\mathbf{x} \mid \mathbf{z}), \tag{21}$$

which can be substituted into (10) to obtain a relaxed version of the joint distribution:

$$\begin{aligned} \hat{p}_\theta(\mathbf{z}, \hat{y}) &= \sum_{\mathbf{x} \in \mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, p(\mathbf{x} \mid \mathbf{z})\, p(\mathbf{z}) \\ &= \sum_{\mathbf{x} \in \mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, p(\mathbf{x}, \mathbf{z}). \end{aligned} \tag{22}$$

Similarly, applying the law of total probability to the relaxed conditional likelihood yields the marginal distribution:

$$\hat{p}_\theta(\hat{y}) = \sum_{\mathbf{x} \in \mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, p(\mathbf{x}). \tag{23}$$

The distributions $p(\mathbf{x}, \mathbf{z})$, $p(\mathbf{x})$, and $p(\mathbf{z})$, which are independent of the model parameters, can be estimated empirically over the entire dataset. Let $\{\mathbf{x}_\ell\}_{\ell=1}^{|\mathcal{X}|}$ and $\{\mathbf{z}_j\}_{j=1}^{|\mathcal{Z}|}$ denote the distinct observed values of the non-sensitive and sensitive features, respectively. Define the empirical joint counts as

$$n(\mathbf{x}_\ell, \mathbf{z}_j) = \sum_{i=1}^{N} \mathbf{1}\left\{\mathbf{x}^{(i)} = \mathbf{x}_\ell,\ \mathbf{z}^{(i)} = \mathbf{z}_j\right\} \tag{24}$$

with marginal counts obtained by summation:

$$n(\mathbf{x}_\ell) = \sum_{j=1}^{|\mathcal{Z}|} n(\mathbf{x}_\ell, \mathbf{z}_j), \tag{25}$$

$$n(\mathbf{z}_j) = \sum_{\ell=1}^{|\mathcal{X}|} n(\mathbf{x}_\ell, \mathbf{z}_j). \tag{26}$$

The corresponding empirical probability estimates are:

$$\hat{p}(\mathbf{x}_\ell, \mathbf{z}_j) = \frac{n(\mathbf{x}_\ell, \mathbf{z}_j)}{N}, \tag{27}$$

$$\hat{p}(\mathbf{x}_\ell) = \frac{n(\mathbf{x}_\ell)}{N}, \tag{28}$$

$$\hat{p}(\mathbf{z}_j) = \frac{n(\mathbf{z}_j)}{N}. \tag{29}$$

Precomputing these estimates on the full dataset prior to training is typically the preferred choice, as it provides statistically stable approximations with reduced variance and minimal sparsity issues. However, in settings where access to the full dataset is constrained or where data statistics evolve dynamically, the same estimation procedure may be applied within each mini-batch. In this case, the batch size B replaces the dataset size N in the above formulas, and the resulting estimates are updated online. Due to limited sample sizes, such batch-level estimates often exhibit higher variance and may suffer from sparsity-related artifacts. To mitigate these issues, a moving average scheme may be employed to combine batch-level estimates over time, smoothing fluctuations while gradually adapting to the evolving data distribution. This approach offers a practical compromise between the statistical stability of offline estimation and the flexibility of online computation, and may be especially beneficial in stochastic optimization regimes.

In this work, the plug-in distributions in (27)–(29) are computed once on the full training set and then kept fixed throughout optimization. Mini-batches are sampled uniformly at random, without explicit stratification by the sensitive attribute $\mathbf{z}$. Consequently, the only batch-dependent terms in the surrogate are the model predictions $\hat{p}_\theta(\hat{y} \mid \mathbf{x})$, while the empirical distributions remain global. The mini-batch-based fairness loss therefore acts as a standard stochastic approximation of the full-data regularization term: sampling noise affects the variance of the gradient estimator, but not the definition of the underlying MI objective.

Incorporating these empirical estimates and the relaxed probability terms into the MI expression in (8), a continuously differentiable surrogate is obtained:

$$\begin{aligned} \hat{I}_\theta(\mathbf{z}, \hat{y}) &= \sum_{\mathbf{z} \in \mathcal{Z}} \sum_{\hat{y} \in \{0,1\}} \hat{p}_\theta(\mathbf{z}, \hat{y}) \log\left(\frac{\hat{p}_\theta(\mathbf{z}, \hat{y})}{\hat{p}(\mathbf{z})\, \hat{p}_\theta(\hat{y})}\right) \\ &= \sum_{\mathbf{z} \in \mathcal{Z}} \sum_{\hat{y} \in \{0,1\}} \left(\sum_{\mathbf{x} \in \mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, \hat{p}(\mathbf{x}, \mathbf{z})\right) \\ &\quad \log\left(\frac{\sum_{\mathbf{x} \in \mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, \hat{p}(\mathbf{x}, \mathbf{z})}{\hat{p}(\mathbf{z}) \sum_{\mathbf{x} \in \mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, \hat{p}(\mathbf{x})}\right). \end{aligned} \tag{30}$$

All components of (30) that depend on θ are continuously differentiable with respect to θ, thereby enabling effective integration of the fairness objective within standard gradient-based training routines.

Approximation guarantees. The Bernoulli relaxation described above can be interpreted as a stochastic channel applied after the deterministic classifier $g_\theta(\mathbf{x})$. The resulting relaxed label $\hat{y}$ is obtained by injecting noise whose distribution depends only on the non-sensitive features $\mathbf{x}$, and not directly on the sensitive attributes $\mathbf{z}$. The plot on the left of Fig. 1 schematically illustrates this computational pipeline, with solid arrows representing deterministic mappings and a dashed arrow representing the stochastic relaxation applied to the classifier output. From a probabilistic standpoint, the key conditional independencies are $\hat{y} \perp \mathbf{z} \mid \mathbf{x}$ and $g_\theta(\mathbf{x}) \perp \mathbf{z} \mid \mathbf{x}$, corresponding to the Markov chains $\mathbf{z} \to \mathbf{x} \to \hat{y}$ and $\mathbf{z} \to \mathbf{x} \to g_\theta(\mathbf{x})$. This emphasizes that the relaxation perturbs the classifier output while preserving the conditional independence structure with respect to $\mathbf{z}$.

As a consequence of this Markov structure, by the data processing inequality, the corresponding relaxed MI $\hat{I}_\theta(\mathbf{z}, \hat{y})$ is guaranteed to be no greater than the true (i.e., non-relaxed) MI, defined here as $I_\theta(\mathbf{z}, \hat{y}) \equiv I_\theta(\mathbf{z}, g_\theta(\mathbf{x}))$:

$$\hat{I}_\theta(\mathbf{z}, \hat{y}) \le I_\theta(\mathbf{z}, g_\theta(\mathbf{x})). \tag{31}$$

Furthermore, the deviation between the true and surrogate MI is bounded as:

$$0 \le I_\theta(\mathbf{z}, g_\theta(\mathbf{x})) - \hat{I}_\theta(\mathbf{z}, \hat{y}) \le \max_{\mathbf{x}} H_b\big(\pi_\theta(\mathbf{x})\big), \tag{32}$$

where $H_b(q) = -q \log q - (1 - q) \log(1 - q)$ denotes the binary entropy.

The upper bound is maximized when $\pi_\theta(\mathbf{x}) = 0.5$, corresponding to maximal predictive uncertainty, and vanishes when $\pi_\theta(\mathbf{x})$ approaches 0 or 1. The characteristic shape of $H_b(\pi)$ is illustrated in the plot on the right of Fig. 1, and reflects how uncertainty controls the tightness of the surrogate.

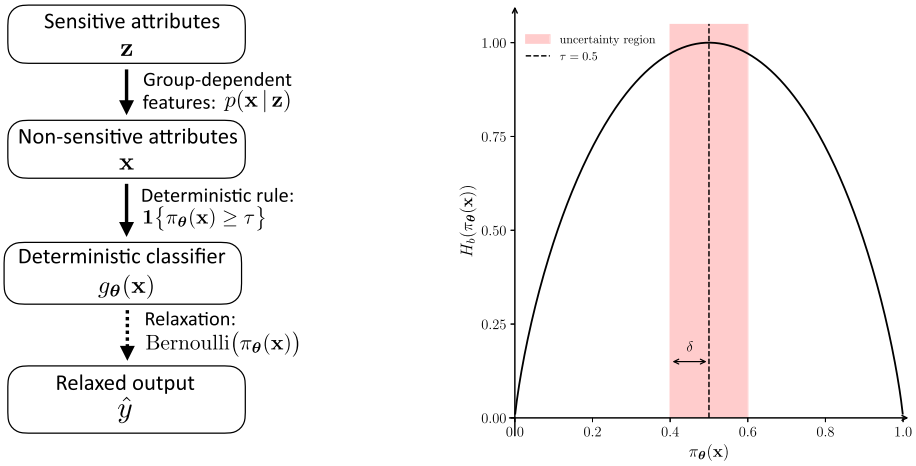


Fig. 1. Schematic structure of the surrogate mutual-information formulation. The plot on the left illustrates the information-flow diagram, highlighting the deterministic mappings (solid arrows) and the stochastic relaxation applied to the classifier output (dashed arrow). The plot on the right shows the binary entropy as a function of $\pi_\theta(\mathbf{x})$, with the high-uncertainty region around 0.5 emphasized to illustrate where the surrogate most strongly deviates from the true mutual information.

More precisely, the tightness of the bound is governed by the distribution of mass near the threshold. Let $\mathcal{A}_\delta := \{\mathbf{x} : |\pi_\theta(\mathbf{x}) - \tau| \leq \delta\}$ with $\delta > 0$ small. When $p(\mathbf{x} \mid \mathbf{z})$ assigns non-negligible probability to $\mathcal{A}_\delta$, the relaxed channel induces maximal label noise in those regions, depressing $\hat{I}_\theta(\mathbf{z}, \hat{y})$ regardless of the dependence produced by the deterministic rule g_θ. Loosening is most pronounced when the near-threshold mass is imbalanced across groups, i.e., $p(\mathcal{A}_\delta \mid \mathbf{z})$ is larger for certain values of $\mathbf{z}$; in that case the surrogate can under-penalize group-localized dependence and tilt the fairness–accuracy trade-off toward utility. Conversely, when uncertainty is low away from the boundary or balanced across groups, $\hat{I}_\theta$ approaches I_θ and the intended trade-off is recovered.

Expectations over $\hat{y}$ in the surrogate MI expression are computed analytically using the values $\pi_\theta(\mathbf{x})$ and $1 - \pi_\theta(\mathbf{x})$, resulting in zero-variance gradients. This property avoids the stochastic noise associated with sampling-based methods and preserves stable optimization dynamics.

As the sample size increases and the Bernoulli relaxation asymptotically approaches deterministic thresholding, the surrogate converges to the true MI, thus ensuring consistency while retaining differentiability throughout training.

Despite its theoretical advantages, the proposed differentiable formulation has two key limitations. First, it relies on frequency-based estimates $\hat{p}(\mathbf{x}_\ell, \mathbf{z}_j)$, $\hat{p}(\mathbf{x}_\ell)$, and $\hat{p}(\mathbf{z}_j)$, which can be sparse or noisy in sample-scarce or high-dimensional settings, potentially making the penalty respond to estimation artifacts rather than genuine statistical dependencies.

The second limitation concerns the assumption that the model's output probability is a well-calibrated measure of confidence. In practice, many discriminative models—and in particular deterministic neural networks— are known to exhibit overconfidence, particularly in ambiguous input regions or during the early stages of training. As a result, miscalibrated predictions can distort the MI estimate, potentially exaggerating or misrepresenting unfairness by assigning undue certainty to spurious associations. It is important to note that this issue specifically affects the surrogate formulation, whereas the original MI in (8) collapses probabilities to hard labels and is thus unaffected by model confidence.

The following subsections introduce extensions to the proposed formulation designed to address these limitations. These enhancements yield a robust, differentiable metric that remains well-suited for integration as a fairness regularization term in gradient-based learning algorithms.

4.2. Kernel smoothing of empirical distributions

The plug-in estimates introduced in (27)–(29) assume that each observed configuration appears with sufficient frequency in the data. However, when the support sets are large, many combinations exhibit low or zero empirical counts. This leads to unreliable, high-variance estimates and undefined logarithmic terms in the surrogate MI expression. Such sparsity is typically driven by the high dimensionality of $\mathbf{x}$. In contrast, the protected attribute vector $\mathbf{z}$ generally has lower dimensionality and therefore may require smoothing only in specific applications. To mitigate sparsity-induced instability while preserving differentiability, a kernel smoothing technique is introduced. This approach enables information sharing across similar configurations of $\mathbf{x}$ and can be trivially extended to $\mathbf{z}$ if needed.

Let $d_H : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ denote a distance function on the discrete input space, such as the Hamming distance. Define a kernel function as a positive, monotonically decreasing function of this distance:

$$K(\mathbf{x}, \mathbf{x}'; \sigma) = f\big(d_H(\mathbf{x}, \mathbf{x}'); \sigma\big), \tag{33}$$

where f is parameterized by a hyperparameter σ, typically interpreted as a bandwidth or scale parameter. Examples include exponential, triangular, or rational quadratic kernels. In practice, the choice of distance d_H, kernel type, and bandwidth σ controls how strongly nearby configurations of $\mathbf{x}$ share statistical strength, trading off local detail against smoother, more stable estimates. Importantly, differentiability of the fairness loss with respect to the model parameters θ is preserved, even if the kernel is not differentiable with respect to $\mathbf{x}$, as long as kernel values remain fixed and independent of θ. The smoothed joint pseudo-counts can be computed offline using the full dataset as follows:

$$\tilde{n}(\mathbf{x}_\ell, \mathbf{z}_j) = \sum_{i=1}^{N} K(\mathbf{x}_\ell, \mathbf{x}^{(i)}; \sigma)\, \mathbf{1}\{\mathbf{z}^{(i)} = \mathbf{z}_j\}, \tag{34}$$

where hard-counting over $\mathbf{x}$ is replaced by kernel-weighted similarity. Marginal pseudo-counts for $\mathbf{x}$ are derived by summing over the observed configurations of $\mathbf{z}$:

$$\tilde{n}(\mathbf{x}_\ell) = \sum_{j=1}^{|\mathcal{Z}|} \tilde{n}(\mathbf{x}_\ell, \mathbf{z}_j). \tag{35}$$

The total kernel-adjusted sample size is given by

$$\tilde{N} = \sum_{\ell, j} \tilde{n}(\mathbf{x}_\ell, \mathbf{z}_j) = \sum_{i, \ell} K(\mathbf{x}_\ell, \mathbf{x}^{(i)}; \sigma), \tag{36}$$

which ensures proper normalization of the pseudo-distribution.

The smoothed joint and marginal distributions are given by:

$$\tilde{p}(\mathbf{x}_\ell, \mathbf{z}_j) = \frac{\tilde{n}(\mathbf{x}_\ell, \mathbf{z}_j)}{\tilde{N}}, \qquad \tilde{p}(\mathbf{x}_\ell) = \frac{\tilde{n}(\mathbf{x}_\ell)}{\tilde{N}}. \tag{37}$$

The marginal $\hat{p}(\mathbf{z}_j)$ is retained as in (29), though smoothing may optionally be applied.

Substituting $\tilde{p}(\mathbf{x}_\ell, \mathbf{z}_j)$ and $\tilde{p}(\mathbf{x}_\ell)$ into (30) yields the kernel-smoothed mutual information surrogate:

$$\tilde{I}_\theta(\mathbf{z}, \hat{y}) = \sum_{\mathbf{z}\in\mathcal{Z}} \sum_{\hat{y}\in\{0,1\}} \left(\sum_{\mathbf{x}\in\mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, \tilde{p}(\mathbf{x}, \mathbf{z}) \right) \log\left(\frac{\sum_{\mathbf{x}\in\mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, \tilde{p}(\mathbf{x}, \mathbf{z})}{\hat{p}(\mathbf{z}) \sum_{\mathbf{x}\in\mathcal{X}} \hat{p}_\theta(\hat{y} \mid \mathbf{x})\, \tilde{p}(\mathbf{x})} \right) \tag{38}$$

All terms remain continuously differentiable in θ, and the positivity of kernel weights ensures numerically stable logarithms. While the same smoothing strategy can be applied online during mini-batch training, this approach incurs significant per-iteration overhead from recomputing distances and normalizing kernels. Offline computation is therefore generally preferred, as it enables precomputing all kernel weights once and amortizing their cost across training iterations, yielding more stable and computationally efficient updates.

To further enhance computational efficiency, kernel methods offer several practical strategies. One such approach is kernel truncation, which reduces the cost of computing smoothed pseudo-counts by limiting the kernel summation to a fixed number of nearest neighbors. This technique approximates the full kernel-weighted sum using only the k most similar data points, thereby lowering computational overhead during offline estimation.

Let $\mathcal{N}_k(\mathbf{x}_\ell) \subset \{1, \ldots, N\}$ denote the set of k nearest neighbors of $\mathbf{x}_\ell$ under the distance function $d_H(\cdot, \cdot)$. The truncated pseudo-counts are then computed as:

$$\tilde{n}_k(\mathbf{x}_\ell, \mathbf{z}_j) = \sum_{i\in\mathcal{N}_k(\mathbf{x}_\ell)} K(\mathbf{x}_\ell, \mathbf{x}^{(i)}; \sigma)\, \mathbf{1}\{\mathbf{z}^{(i)} = \mathbf{z}_j\}. \tag{39}$$

The hyperparameter k controls the trade-off between computational efficiency and statistical robustness: small values accelerate computation but may lead to unstable estimates for rare or outlying configurations of $\mathbf{x}$, while large values increase stability at the expense of higher cost. In practice, moderate values of k often provide a good balance, especially when combined with safeguards against zero counts. In low-density regions, truncated neighborhoods may yield insufficient kernel weight, causing pseudo-counts to approach zero and reducing the robustness of the surrogate. To mitigate this risk, positivity safeguards are introduced to ensure that all pseudo-counts remain strictly positive.

One safeguard consists of explicitly including the self-match $\mathbf{x}^{(i)} = \mathbf{x}_\ell$ in $\mathcal{N}_k(\mathbf{x}_\ell)$. An alternative strategy involves applying a kernel floor, defined as:

$$K(\mathbf{x}, \mathbf{x}'; \sigma) \leftarrow \max\left(K(\mathbf{x}, \mathbf{x}'; \sigma),\, \varepsilon\right), \quad \varepsilon > 0. \tag{40}$$

These techniques guarantee that kernel-weighted pseudo-counts are bounded away from zero, preserving numerical stability, ensuring the well-defined nature of the MI surrogate under truncation, and maintaining compatibility with gradient-based optimization frameworks.

4.3. Bayesian approximation via monte carlo dropout

To address model overconfidence and account for epistemic uncertainty, a Bayesian approximation is introduced via Monte Carlo (MC) dropout to enhance the robustness of the proposed fairness metric. This technique treats the classifier as a stochastic function by enabling dropout layers during both training and inference. In this setting, a binary mask is sampled at each pass to induce stochasticity:

$$\mathbf{m} = (m_1, \ldots, m_d) \in \{0, 1\}^d, \quad m_j \sim \text{Bernoulli}(1 - p_{\text{drop}}). \tag{41}$$

This mask is applied independently to each of the d trainable parameters collected in θ. Define the joint variational posterior over weights $\mathbf{w}$ and dropout mask $\mathbf{m}$ as

$$q_\theta(\mathbf{w}, \mathbf{m}) = \left[\prod_{j=1}^{d} (1 - p_{\text{drop}})^{m_j}\, p_{\text{drop}}^{1-m_j} \right] \delta\big(\mathbf{w} - \theta \odot \mathbf{m}\big), \tag{42}$$

where the product term gives the probability mass of the sampled mask $\mathbf{m}$ and the Dirac delta enforces the deterministic mapping $\mathbf{w} = \theta \odot \mathbf{m}$.

By marginalizing out the dropout mask, the variational posterior over weights becomes:

$$q_\theta(\mathbf{w}) = \sum_{\mathbf{m}\in\{0,1\}^d} \left[\prod_{j=1}^{d} (1 - p_{\text{drop}})^{m_j}\, p_{\text{drop}}^{1-m_j} \right] \delta\big(\mathbf{w} - \theta \odot \mathbf{m}\big). \tag{43}$$

This expression describes a discrete mixture of Dirac measures, each corresponding to a subnetwork defined by a specific dropout configuration. Each MC sample drawn from $q_\theta(\mathbf{w})$ corresponds to a unique mask realization and its associated subnetwork. Under this formulation, the model behaves as a stochastic function, with each forward pass generating predictions from a different subnetwork.

During training, dropout is selectively enabled only in the computation of the fairness loss. The utility loss is evaluated with dropout disabled, ensuring that the optimization is performed with respect to the deterministic mean network. This choice provides low-variance, unbiased gradient estimates for the log-likelihood component of the objective. In contrast, the fairness loss requires accounting for the full posterior predictive distribution induced by dropout. To that end, T stochastic forward passes are performed per input $\mathbf{x}$, producing a collection of predictive probabilities $\{p_\theta^{(t)}(y = 1 \mid \mathbf{x})\}_{t=1}^{T}$. These samples represent draws from the approximate posterior predictive distribution and capture epistemic uncertainty. In this construction, the stochasticity induced by MC dropout defines a distribution over subnetworks; the fairness objective is therefore evaluated as the expectation of the kernel-smoothed MI surrogate with respect to this distribution.

For each sample, the smoothed MI surrogate $\tilde{I}_\theta(\mathbf{z}, \hat{y}^{(t)})$ is computed using (38), where $\hat{y}^{(t)}$ denotes the corresponding stochastic label realization. The final Bayesian term is computed via MC averaging:

$$\tilde{I}_\theta^{\text{MC}}(\mathbf{z}, \hat{y}) = \frac{1}{T} \sum_{t=1}^{T} \tilde{I}_\theta(\mathbf{z}, \hat{y}^{(t)}). \tag{44}$$

This estimator makes explicit that MC dropout contributes to the fairness loss solely through the averaging in (44). No term based on the sample variance of the predictive probabilities is introduced; predictive variability affects the loss only indirectly, through its influence on the individual surrogate values $\tilde{I}_\theta(\mathbf{z}, \hat{y}^{(t)})$ entering the MC expectation. This expression defines the final form of the differentiable, uncertainty-aware MI surrogate, which is used as the fairness loss term

$$L_{\text{f}} = \tilde{I}_\theta^{\text{MC}} \tag{45}$$

within the composite training objective (4). The hyperparameter T directly controls the bias–variance trade-off of the estimator in (44): small values of T reduce computational overhead but increase the variance of the fairness loss, whereas large values stabilize the estimate at the cost of additional forward passes. In practice, moderate values (e.g., T between 5 and 20) often suffice to capture predictive variability without

incurring prohibitive cost. The dropout rate p_{drop} also interacts with this trade-off: higher dropout induces stronger posterior variability, which enhances robustness to overconfidence but may lead to noisier fairness penalties if T is too small. The surrogate remains fully differentiable with respect to θ, as the stochasticity introduced by dropout lies outside the computational graph. To improve numerical stability and avoid unnecessary memory overhead when T is large, the MC average in (44) is implemented using a running-mean update:

$$L_{\text{f}}^{(t)} = \frac{t-1}{t} L_{\text{f}}^{(t-1)} + \frac{1}{t} \tilde{I}_\theta(\mathbf{z}, \hat{y}^{(t)}), \tag{46}$$

which keeps the fairness loss L_{f} on the same scale as the total loss at each iteration and introduces no additional computational cost. The expression in (44) is an unbiased estimator of the expectation

$$\mathbb{E}_{\mathbf{w}\sim q_\theta(\mathbf{w})}\left[\tilde{I}_\theta(\mathbf{z}, \hat{y}^{(t_{\mathbf{w}})})\right], \tag{47}$$

where $\tilde{I}_\theta^{(t_{\mathbf{w}})}$ denotes the MI surrogate computed using predictions from a particular dropout-induced weight realization $\mathbf{w}$. Owing to the convexity of the KL divergence and Jensen's inequality, this expectation upper bounds the surrogate MI of the averaged predictive distribution:

$$\mathbb{E}_{\mathbf{w}}\left[\tilde{I}_\theta(\mathbf{z}, \hat{y}^{(t_{\mathbf{w}})})\right] \geq \tilde{I}_\theta\left(\mathbf{z}, \mathbb{E}_{\mathbf{w}}[\hat{y}^{\text{MC}}]\right), \tag{48}$$

where $\hat{y}^{\text{MC}}$ denotes the label assigned by thresholding the average predictive probability across MC samples. As a result, minimizing (44) encourages reduction of dependence between $\mathbf{z}$ and the final predictions, even under uncertainty. The resulting regularizer is conservative—it may overestimate dependence, but never underestimate it—thus providing robustness guarantees under uncertainty. Intuitively, regions where the posterior predictive distribution is more dispersed across dropout samples exhibit higher conditional entropy and therefore contribute less to the expectation in (44); this attenuation arises intrinsically from the MI surrogate and does not rely on any additional variance-based penalty. Moreover, this formulation avoids over-penalizing uncertain predictions: near the decision boundary, the entropy of the predictive distribution increases due to dropout-induced variability, which reduces MI. Therefore, the fairness penalty is naturally attenuated in ambiguous regions, allowing the model to remain flexible where confidence is low while enforcing fairness more stringently where the model is confident.

The complete training pipeline is summarized in Algorithm 1, which details (i) the required inputs and hyperparameters, (ii) the offline kernel-based computation of empirical pseudo-counts, and (iii) the stochastic gradient-based optimization combining utility and fairness losses. A complementary schematic overview is provided in Fig. 2. To ensure consistency between training and deployment, dropout remains active at inference time. Final predictions are obtained by averaging MC samples,

$$\bar{\pi}_\theta(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} p_\theta^{(t)}(y = 1 \mid \mathbf{x}) \tag{49}$$

followed by thresholding. This inference strategy ensures that the deployed model adheres to the same stochastic prediction rule used during training for fairness optimization. A mild mismatch arises with respect to the utility loss, which is optimized under a deterministic (dropout-disabled) predictor. This introduces a slight bias in the training objective, as the deterministic model underestimates the expected log-loss of the deployed ensemble. The bias is typically small in practice, as predictive distributions produced by different dropout masks tend to be highly correlated. If exact alignment is required, the utility loss can be replaced with its MC estimate using the same set of stochastic forward passes. In this work, the deterministic utility formulation is retained to improve gradient stability and computational efficiency, without substantially affecting performance.

Algorithm 1 Training pipeline with differentiable, uncertainty-aware fairness regularization.

Require: Training dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}, y^{(i)})\}_{i=1}^{N}$
Neural model architecture $p_\theta(y \mid \mathbf{x})$
Dropout rate p_{drop}
Kernel type $K(\cdot,\cdot;\sigma)$ and scale hyperparameter σ
Fairness regularization weight α, learning rate η
Batch size B, number of epochs E
Number of MC samples T
Number of neighbors k (for kernel truncation)

Offline Preprocessing:
  Compute empirical count $n(\mathbf{z}_j)$
  $\forall\, \mathbf{x}_{\ell'} \in \mathcal{X}$, find top-$k$ nearest neighbors under $d_H(\cdot,\cdot)$
  Compute pseudo-counts $\tilde{n}_k(\mathbf{x}_{\ell'}, \mathbf{z}_j)$
  Normalize to obtain $\tilde{p}(\mathbf{x}_{\ell'}, \mathbf{z}_j)$, $\tilde{p}(\mathbf{x}_{\ell'})$, and $\hat{p}(\mathbf{z}_j)$
Initialize:
  Model parameters θ
  Optimizer state
for epoch = 1 **to** E **do**
  for each mini-batch $\mathcal{B} \subset \mathcal{D}$ **do**
    Utility Loss: compute $L_{\text{u}}(\theta)$ on $\mathcal{B}$ (dropout off)
    Fairness Loss: initialize $L_{\text{f}}(\theta) \leftarrow 0$
    for $t = 1$ **to** T **do**
      Sample mask $\mathbf{m}^{(t)} \sim \text{Bernoulli}\,(1 - p_{\text{drop}})^{\otimes d}$
      Forward pass to get $\pi_\theta^{(t)}(\mathbf{x}^{(i)})$ for all $(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}, y^{(i)}) \in \mathcal{B}$
      Compute $\tilde{I}_\theta^{(t)} := \tilde{I}_\theta(\mathbf{z}, \hat{y}^{(t)})$
      Running average: $L_{\text{f}}(\theta) \leftarrow \frac{t-1}{t} L_{\text{f}}(\theta) + \frac{1}{t} \tilde{I}_\theta^{(t)}$
    end for
    Total Loss: $L(\theta) \leftarrow (1-\alpha)\, L_{\text{u}}(\theta) + \alpha\, L_{\text{f}}(\theta)$
    Gradient Step: backpropagate and update θ
  end for
end for
Return: trained parameters θ^*

4.4. Scalability and complexity analysis

This subsection analyzes the computational complexity of the full training pipeline, considering both offline preprocessing and online optimization. The analysis focuses on scalability with respect to the training dataset size N, and compares the method to alternative gradient-free in-processing approaches. The discussion is architecture-agnostic: it applies to any differentiable probabilistic classifier $p_\theta(y \mid \mathbf{x})$, with model-specific details absorbed into per-batch forward/backward costs defined below.

Prior to training, the algorithm computes empirical estimates of $\hat{p}(\mathbf{z})$, $\tilde{p}(\mathbf{x}, \mathbf{z})$, and $\tilde{p}(\mathbf{x})$ using kernel smoothing over the dataset $\mathcal{D}$. When no truncation is applied, the all-pairs kernel distances require $O(N^2)$ operations. However, with k-nearest neighbor truncation, the cost is reduced to $O(Nk)$ under fixed bandwidth σ and approximate neighbor search. These kernel-weighted pseudo-counts are computed once and cached, resulting in a one-time preprocessing cost that scales near-linearly in N when $k \ll N$ and is easily parallelizable.

During training, stochastic gradient descent is applied over E epochs with mini-batches of size B. For each batch, the utility loss is computed once with dropout disabled, while the fairness loss requires T stochastic forward passes under different dropout masks (or, more generally, different stochastic subnetworks). Each MC sample evaluates the MI surrogate in (38) using the precomputed distributions. The per-batch computational cost is $O\big((1+T)(C_{\text{fwd}} + C_{\text{bwd}})\big)$, where C_{fwd} and C_{bwd} are the costs of a forward and backward pass, respectively.

Aggregating across all batches and epochs yields an overall worst-case training complexity of

$$O\big(EN(1+T)(C_{\text{fwd}} + C_{\text{bwd}})\big), \tag{50}$$

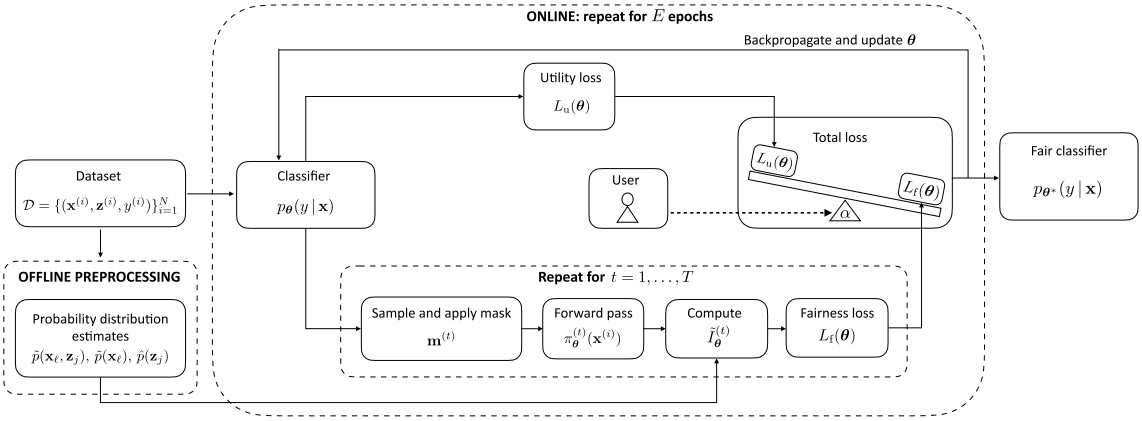


Fig. 2. Pipeline schematic with offline and online stages; the fairness weight α is user-adjustable to balance bias and accuracy. Batch-level implementation details are omitted for brevity and clarity.

which is linear in both the dataset size N and the number of MC passes T. The constants C_{fwd} and C_{bwd} depend only on model depth and do not scale with N, while kernel truncation limits the dependence on input dimensionality to $O(k)$. This expression represents the worst-case asymptotic bound for the proposed training pipeline.

Compared to gradient-free optimization, the proposed approach offers substantial advantages. Using the non-differentiable MI metric directly would necessitate black-box methods such as the genetic algorithm (GA) [41], which operates by evaluating the model over the entire training dataset for each individual in the population. For a GA with population size P and G generations, the total computational cost is $O(PGN)$ forward passes, each involving a full traversal of the dataset (with a per-pass cost that still depends on the chosen architecture). This results in a number of full evaluations that is typically several orders of magnitude larger than the number of training steps required by standard gradient-based methods. Additionally, gradient-free methods scale poorly with the number of parameters d and lack the directional information provided by backpropagation, often resulting in slow convergence, particularly in high-dimensional spaces.

In contrast, the proposed training strategy scales linearly with N and T, introduces only a constant-time overhead relative to standard training, and maintains compatibility with GPU-accelerated autodiff frameworks.

From a practical standpoint, however, the effective number of epochs often saturates or decreases with N due to early stopping, and mini-batch parallelization amortizes part of the per-batch overhead. Consequently, the effective runtime may grow at a rate slightly below the theoretical linear bound. Such deviations are implementation-dependent and reflect optimization and hardware efficiencies rather than a tighter asymptotic property.

5. Experimental results

This section presents an experimental evaluation of the proposed fairness-aware learning strategy, which incorporates a differentiable surrogate of MI into the training objective of a nonlinear binary classifier. As detailed in the previous section, the surrogate is fully differentiable, robust to uncertainty through MC dropout, and stabilized by kernel-based smoothing techniques—features that enable effective and scalable fairness regularization in gradient-based optimization pipelines. The goal of the experiments is to assess the practical utility of this surrogate, particularly in scenarios where nonlinear models are required to capture complex data dependencies and offer greater flexibility than traditional approaches.

Two datasets are considered for empirical validation: a synthetic dataset and a real-world dataset. The majority of the experiments are conducted in the synthetic setting, which offers full control over the generative dependencies between input, sensitive, and target variables. This flexibility enables rigorous testing of the method's fairness behavior and generalization properties, as well as specialized evaluations—such as counterfactual fairness analysis and scalability assessment via controlled variation in sample size.

In addition, the proposed approach is applied to a real-world dataset to assess its practical utility in realistic settings that reflect complex socioeconomic structures. The chosen dataset is widely adopted in the fairness literature, enabling comparisons against existing methods and serving as a benchmark for empirical performance.

The remainder of this section is organized as follows. The experimental setup is first detailed—covering datasets, baseline models, hyperparameter choices, and evaluation metrics. Next, synthetic-data results are reported, including counterfactual and scalability analyses and a sensitivity study of key hyperparameters. Finally, results on real-world datasets are presented.

5.1. General experimental settings

The synthetic dataset used in this study was generated following the two-step procedure introduced by Barbierato et al. [42], which leverages Structural Equation Modeling (SEM) to construct data with explicitly controlled MI between specified attributes. This approach facilitates the creation of a synthetic environment with indirect bias patterns that can be precisely tuned and studied.

The generated dataset comprises $n = 50{,}000$ samples, each characterized by:

- A binary target variable y;
- Ten discrete non-sensitive attributes $x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}$;
- Two discrete sensitive attributes z_1, z_2.

A key design feature of this synthetic setting is the introduction of indirect bias: the sensitive attributes z_1 and z_2 are not directly used for target generation but are correlated with the non-sensitive inputs x_1 and x_2, respectively. Meanwhile, the target variable y is generated solely based on the non-sensitive attributes. This structure allows for the presence of bias through statistical dependency, without explicit use of sensitive information in the classification rule. The characteristics of the variables are summarized in Table 1, and the underlying dependency structure is visualized in the Bayesian network presented in Fig. 3, which highlights how the sensitive attributes influence the output indirectly through their correlations with specific non-sensitive features.

In addition to these primary dependencies, a sparse, context-specific interaction is introduced between x_2 and x_8, which becomes active only when $x_5 \in S = \{5, 12, 19\}$. This subset corresponds to approximately

Table 1
Attribute summary of the synthetic dataset, including data type, number of unique values, and assigned role (sensitive, non-sensitive, or target).

Attributes	Type	Values	Role
z_1	binary	2	sensitive
z_2	ordered categorical	3	sensitive
x_1	binary	2	non-sensitive
x_2	ordered categorical	10	non-sensitive
x_3	binary	2	non-sensitive
x_4	ordered categorical	15	non-sensitive
x_5, x_6, x_7, x_8	ordered categorical	20	non-sensitive
x_9, x_{10}	ordered categorical	100	non-sensitive
y	binary	2	target

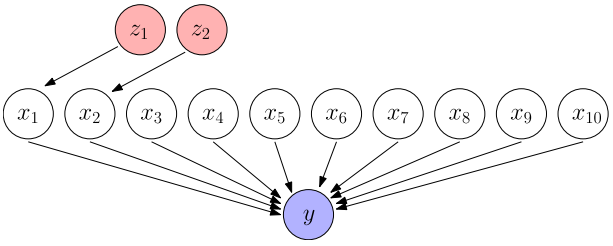


Fig. 3. Bayesian network structure of the synthetic dataset used in this study. Nodes represent dataset attributes, with red indicating sensitive features, blue the target variable, and white the non-sensitive features. Directed edges denote conditional dependencies.

15 % of the marginal distribution of x_5, ensuring that the effect remains rare but non-negligible. Under these conditions, the conditional distribution of x_8 is obtained by applying an exponential tilt to its baseline distribution, with a small tilting coefficient $\lambda = 0.08$. This construction induces a weak, non-local association without altering the global dependency structure.

The data generation process can be briefly outlined as follows:

1. *Gradient-free optimization (Independent estimation):* The desired MI levels between selected attribute pairs are first specified. To achieve these targets, correlation coefficients are tuned through a direct search over the interval $[-1, 1]$, where MI is numerically computed for each candidate. The value minimizing the deviation from the target MI is selected. Due to the lack of a closed-form expression linking correlation and MI, this step is implemented using a gradient-free method, specifically the Nelder-Mead algorithm.
2. *Gradient-based optimization (Joint parameter fitting):* Once the optimal correlation coefficients are identified, the SEM parameters—including linear coefficients and noise variances—are jointly optimized using gradient-based techniques to ensure that the resulting covariance structure aligns with the desired correlations.

Following the SEM fitting phase, the binary target y is generated through a logistic regression model incorporating both linear and nonlinear terms. Specifically, y depends on the main linear effects of the non-sensitive attributes and a second-order interaction term defined as $x_1x_2 + x_1x_6 + x_3^2 - x_4^2 + x_7x_8 + x_9^2 + x_{10}^2$, thus introducing nonlinearity into the classification task. The regression coefficients, derived from the SEM parameter optimization, are set as follows: $\gamma_1 = 0.75$, $\gamma_2 = 0.6$, $\gamma_3 = 0.25$, $\gamma_4 = 0.4$ for the respective input variables, and $\gamma_5 = 1.0$ for the interaction term.

For the second part of the experimental evaluation, the 2001 Dutch Census dataset [43] was selected. This dataset is widely used in fairness literature and provides a comprehensive snapshot of socio-economic and demographic characteristics of the Dutch population at the time of the 2001 census. The original version contains 60,420 samples and 12 attributes. For the purpose of this study, the feature space was reduced to the five most informative attributes in terms of mutual information with respect to the binary target variable, which indicates occupational status.

The selected attributes were preprocessed to ensure a clean, concise, and numerically balanced dataset, without compromising the underlying causal and statistical structures relevant to fairness analysis. Preprocessing was performed as follows:

- **Age**: The original numerical values were binned into discrete age groups: 15–19, 20–24, 25–34, 35–49, and 50–59. To address data imbalance, the last two groups were merged, resulting in four final bins, which were treated as an ordered categorical variable and mapped to integer codes.
- **Current economic activity ("cur_eco_activity")**: This variable was reclassified into four broad sectors—*Public Sector*, *Business and financial*, *Trade and Services*, and *Primary/Industry*—and subsequently one-hot encoded.
- **Economic status**: This categorical variable was directly one-hot encoded.

The structure of the resulting dataset is summarized in Table 2, and the underlying dependency relations among the variables are depicted in the Bayesian network shown in Fig. 4.

Table 2
Summary of attribute characteristics for the 2001 Dutch Census dataset after preprocessing. Each attribute is annotated with its type, number of unique values, role in the classification task (sensitive, non-sensitive, or target), and a brief description.

Attributes	Type	Values	Role	Description
sex	binary	2	sensitive	The biological sex of the person {Male; Female}
age	ordered categorical	4	sensitive	The age group of the person {15–19; 20–24; 25–34; 35–59}
edu_level	ordered categorical	6	non-sensitive	The person's education level, ranging from 0 (lowest) to 5 (highest)
economic_status	categorical	3	non-sensitive	The person's economic status (class of worker) {Salary worker; Self-employed; Unemployed}
cur_eco_activity	categorical	4	non-sensitive	The current economic activity {Public Sector; Business and Financial; Trade and Services; Primary/Industry}
occupation	binary	2	target	The person's occupation {Low-level; High-level}

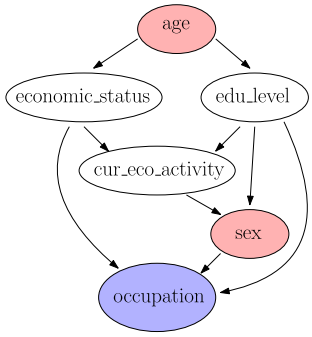


Fig. 4. Bayesian network structure of the preprocessed 2001 Dutch Census dataset. Nodes represent dataset attributes, with red indicating sensitive features, blue the target variable, and white the non-sensitive features. Directed edges denote conditional dependencies.

Although the attribute `sex` is commonly used as the primary protected attribute in fairness literature, in this study, the variable `age` is also considered a sensitive attribute. This choice introduces additional complexity to the fairness evaluation framework, allowing the analysis to account for multiple sensitive features. Importantly, while `sex` directly influences the target, `age` contributes to unfairness through indirect statistical dependencies. From a practical perspective, this inclusion is well-justified, as age can often be a source of unjustified disparities in treatment, even when qualifications or competencies are comparable.

All experiments presented in this work, both on synthetic and real datasets, follow a consistent evaluation protocol. Each dataset is partitioned into training, validation, and testing subsets using a fixed split ratio of 60 %, 20 %, and 20 %, respectively.

The primary nonlinear backbone within the proposed framework is a Multi-Layer Perceptron (MLP) with two hidden layers, configured as 64 and 32 units. To demonstrate architectural generality, two additional variants were considered: a linear model based on logistic regression with an ℓ_2 (ridge) penalty set to 0.1, and an FT-Transformer, a state-of-the-art architecture for tabular data [44]. The FT-Transformer tokenizes each feature and applies self-attention over the resulting sequence, enabling the model to capture higher-order and context-specific feature interactions while remaining permutation-agnostic with respect to the input ordering and naturally handling heterogeneous attributes. The adopted FT-Transformer uses $d_{\text{token}} = 64$ with a learnable feature tokenizer, $N_{\text{blocks}} = 3$ encoder blocks, $N_{\text{heads}} = 4$ attention heads, GELU activations, pre-norm LayerNorm, an expansion factor of 2 in the feed-forward sublayers ($d_{\text{ff}} = 128$), a classification token for aggregation, attention and residual dropout of 0.10, token (embedding) dropout of 0.05, and stochastic depth of 0.05.

Training is performed using stochastic gradient descent with a learning rate η set to 0.05, a batch size of 256, and a maximum of 200 epochs. Early stopping is applied to prevent overfitting, with a patience of 40 epochs and a tolerance threshold of 1 %.

The training objective consists of two components: a utility loss L_{u}, instantiated as BCE, which optimizes the model's predictive accuracy, and a fairness loss L_{f}, defined using the proposed differentiable surrogate of MI. The latter acts as a proxy for classifier bias, quantified in this work as the statistical dependence between sensitive attributes $\mathbf{z}$ and predicted labels $\hat{y}$, formally measured by the MI $I_\theta(\mathbf{z}, \hat{y})$. In all experiments, the sensitive attribute vector $\mathbf{z}$ is multidimensional, and the fairness objective treats each of its components equally, without prioritizing any individual dimension.

The implementation integrates offline kernel smoothing of the probability distributions associated with the non-sensitive features $\mathbf{x}$ and MC dropout to improve robustness and uncertainty calibration, as detailed in Section 4. The kernel function $K(\cdot, \cdot; \sigma)$ used for smoothing is exponential, with the scale hyperparameter σ set to 1. To mitigate the computational cost of all-pairs similarity calculations, kernel truncation is applied using k-nearest neighbors with $k = 8$, computed via approximate nearest-neighbor search. The dropout probability is fixed at $p_{\text{drop}} = 0.2$, and $T = 10$ MC samples are drawn per mini-batch to compute the uncertainty-aware fairness objective.

To analyze the impact of fairness regularization, experiments are conducted across multiple values of the regularization parameter α, ranging from 0 to 0.8. Although the theoretical domain of α spans the interval $[0, 1]$, values approaching 1 were excluded from practical evaluation. This decision is motivated by the observation that excessive emphasis on the fairness term leads to over-regularization, severely impairing the model's ability to capture the underlying data distribution. Under such conditions, the classifier often produces degenerate outputs, becoming nearly invariant with respect to the target labels, due to the dominance of the fairness constraint over the utility loss. Therefore, α is limited to a subrange where the model maintains sufficient predictive capacity while still addressing fairness concerns.

The proposed pipeline is also compared against standard bias-mitigation techniques, specifically two classical pre-processing methods—massaging and reweighting [11]—a post-processing method known as reject option classification (ROC) [45]—and a widely adopted in-processing approach, adversarial debiasing [46].

For the massaging strategy, a modified implementation is adopted, differing from the original in its use of a ratio-based fairness metric instead of statistical parity difference. The number of label modifications is computed proportionally to the dataset's initial unfairness, and a priority-based sorting strategy is employed to minimize classification disruption. Specifically, low-confidence privileged positives and high-confidence disadvantaged negatives are targeted for relabeling.

The reweighting approach estimates the expected frequencies of sensitive attribute–label pairs under the assumption of independence, assigning sample weights proportional to the ratio between expected and observed joint probabilities. This weighting scheme is designed to reduce bias by emphasizing underrepresented group–label combinations while maintaining the data distribution. The resulting weights are integrated into the loss function during training.

The ROC approach operates on the calibrated prediction scores produced by the trained classifier, leaving the training pipeline unchanged. At test time, instances whose predicted scores fall within a symmetric uncertainty band $[\tau_{\text{low}}, \tau_{\text{high}}]$ around the classification threshold are identified as candidates for corrective relabeling. Within this reject region, predictions are reassigned to favor the disadvantaged group—privileged positives are flipped to negatives, and disadvantaged negatives to positives—thereby reducing disparate impact while preserving high-confidence decisions. The adopted configuration employs a reject-band width of $\delta_{\text{ROC}} = 0.15$ and a group-specific flip rate coefficient of $\gamma_{\text{ROC}} = 0.6$.

For the massaging, reweighting, and ROC strategies, an MLP classifier with the dataset-specific architecture is employed, without any additional in-processing fairness regularization (i.e., $\alpha = 0.0$).

For adversarial debiasing, an in-processing minimax configuration following [46] is adopted, with the predictor instantiated as an MLP identical in architecture to that used for the proposed method. The adversary is a shallow MLP comprising two hidden layers with 32 and 16 units, respectively, ReLU activations, LayerNorm between layers, and a dropout rate of 0.20; it branches into two heads (one per sensitive attribute), each a binary softmax trained with cross-entropy. Coupling to the predictor is implemented via a gradient-reversal layer parameterized by λ_{adv}. Training alternates one predictor update with one adversary update per mini-batch, using the same optimizer, learning rate, early-stopping rule, and batching strategy as the other experiments; gradients are clipped at 1.0, and label smoothing of 0.05 is applied to stabilize the adversary. To trace accuracy–bias frontiers, λ_{adv} is varied within the interval $[0, 0.8]$.

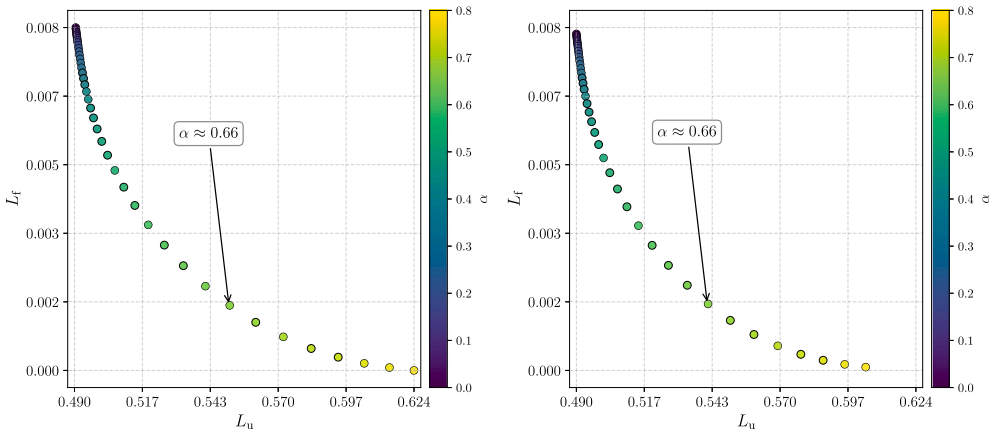


Fig. 5. Trade-off between utility loss L_u and fairness loss L_f at training time across regularization values α for the proposed nonlinear classifier in the two considered variants: MLP (left) and FT-Transformer (right). Each point corresponds to the final loss values obtained for a specific α, with color indicating its magnitude.

All hyperparameter values—including model architectures, optimization settings, kernel parameters, dropout rate, and number of MC samples—were selected based on validation performance in the synthetic setting using a grid search over a diverse set of configurations. The same values were reused for the real-world dataset to assess cross-domain robustness, with two targeted adjustments to reflect its lower effective complexity: the MLP hidden-layer sizes were set to $(32, 32)$ (instead of $(64, 32)$ in the synthetic experiments), and the FT-Transformer dropout on attention and residual connections was increased to 0.20 (from 0.10). All other settings remained unchanged.

For clarity and reproducibility, the hyperparameter configurations adopted for the proposed method are concisely summarized in Table A.1 of Appendix A.

The evaluation focuses primarily on bias–accuracy trade-offs obtained by adjusting the relevant hyperparameters of each approach. Leveraging the synthetic framework, additional analyses are conducted: counterfactual fairness, computational complexity, and a sensitivity study of key hyperparameters.

All experiments were performed on a MacBook Pro equipped with an Apple M3 Pro chip and 18 GB of unified memory, running macOS 14.6.1. The implementation was developed in Python, utilizing the PyTorch framework.

5.2. Synthetic dataset experiments

The impact of the regularization parameter α on the training behavior of the proposed approach is illustrated in Fig. 5, which depicts the trade-off between the utility loss and the fairness loss for the MLP (left) and FT-Transformer (right) architectures. As expected, due to the convex formulation of the cost function, improvements in one loss component generally correspond to a decline in the other. Interestingly, the fairness loss exhibits an exponential decay as the utility loss increases, corresponding to higher values of α. Notably, the plots suggest the presence of a "knee" point at approximately $\alpha \approx 0.66$ in both cases, up to which the fairness loss decreases by approximately 79 %, while the utility loss increases by only about 13 % for the MLP case and 10 % for the FT-Transformer case, indicating favorable trade-off regions.

This observation is corroborated by results on the test set, where the generalization performance of the proposed nonlinear approaches is benchmarked against the linear counterpart and additional bias-mitigation strategies.

The comparative results are presented in Fig. 6, where each model's bias (x-axis) is plotted against its accuracy (y-axis). For the in-processing methods, multiple trade-off points are shown, corresponding to different values of α (for the proposed pipeline) and to the scaling coefficient λ_{adv} (for adversarial debiasing).

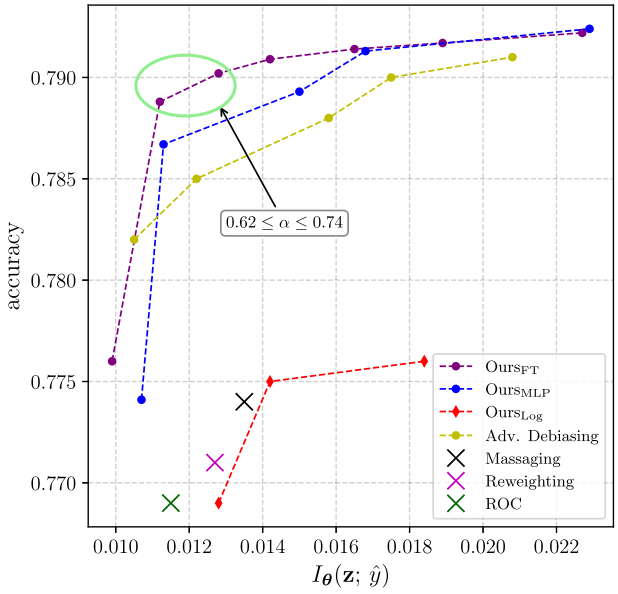


Fig. 6. Bias–accuracy trade-off on the synthetic test set for the proposed in-processing method with three backbones—MLP and FT-Transformer (nonlinear) and logistic (linear)—plus an in-processing adversarial debiasing model, two pre-processing techniques (massaging, reweighting) and a post-processing approach (ROC). Each point represents test performance for a given value of the regularization parameter α; for adversarial debiasing, each point corresponds to a value of the adversarial scaling coefficient λ_{adv}; overlapping points indicate repeated results for different regularization coefficient values. In its nonlinear variants, the proposed method traces a consistently stronger bias–accuracy frontier than the alternative approaches. The green-circled region highlights α values in the range $[0.62,\ 0.74]$, where a favorable balance between bias mitigation and accuracy emerges, particularly pronounced for the FT-Transformer.

As anticipated, an increase in fairness (i.e., reduced bias) is generally accompanied by a drop in accuracy. Nevertheless, the proposed nonlinear strategies consistently outperform the baselines, offering improved performance and greater flexibility. In particular, for α ranging from 0 to 0.8, the FT-Transformer-based and MLP-based approaches yielded seven and five bias–accuracy trade-offs, respectively, compared with

Table 3
Comparison of test-time bias and accuracy on the synthetic dataset for the proposed approach, in all its variants, and the adversarial debiasing method, evaluated across varying values of the regularization parameter α (λ_{adv} for the adversarial debiasing). Results from two pre-processing baselines—massaging and reweighting—and the ROC post-processing approach are included as well. For each method, percentage changes are reported in parentheses relative to the same method with no bias mitigation ($\alpha = 0$). For the pre-processing and post-processing strategies, the percentage changes are relative to the proposed approach under the MLP backbone and $\alpha = 0$, as they share the same model architecture.

α	Ours_{MLP}		Ours_{FT}		Ours_{Log}		λ_{adv}	Adv. Debiasing	
	Bias	**Accuracy**	**Bias**	**Accuracy**	**Bias**	**Accuracy**		**Bias**	**Accuracy**
0.0	0.0229	0.7924	0.0227	0.7922	0.7760	0.0184	0.0	0.0208	0.7910
0.1	0.0168	0.7913	0.0189	0.7917	0.7760	0.0184	0.1	0.0175	0.7900
	(−26.64 %)	(−0.14 %)	(−16.74 %)	(−0.06 %)	(0.00 %)	(0.00 %)		(−15.87 %)	(−0.13 %)
0.2	0.0150	0.7893	0.0165	0.7914	0.7760	0.0184	0.2	0.0158	0.7880
	(−34.50 %)	(−0.39 %)	(−27.31 %)	(−0.10 %)	(0.00 %)	(0.00 %)		(−24.04 %)	(−0.38 %)
0.3	0.0150	0.7893	0.0142	0.7909	0.7760	0.0184	0.3	0.0158	0.7880
	(−34.50 %)	(−0.39 %)	(−37.44 %)	(−0.16 %)	(0.00 %)	(0.00 %)		(−24.04 %)	(−0.38 %)
0.4	0.0150	0.7893	0.0142	0.7909	0.7760	0.0184	0.4	0.0158	0.7880
	(−34.50 %)	(−0.39 %)	(−37.44 %)	(−0.16 %)	(0.00 %)	(0.00 %)		(−24.04 %)	(−0.38 %)
0.5	0.0150	0.7893	0.0142	0.7909	0.7760	0.0184	0.5	0.0158	0.7880
	(−34.50 %)	(−0.39 %)	(−37.44 %)	(−0.16 %)	(0.00 %)	(0.00 %)		(−24.04 %)	(−0.38 %)
0.6	0.0150	0.7893	0.0128	0.7902	0.7750	0.0142	0.6	0.0158	0.7880
	(−34.50 %)	(−0.39 %)	(−43.61 %)	(−0.25 %)	(−0.13 %)	(−22.83 %)		(−24.04 %)	(−0.38 %)
0.7	**0.0113**	**0.7867**	**0.0112**	**0.7888**	0.7690	0.0128	0.7	0.0122	0.7850
	(−50.66 %)	**(−0.72 %)**	**(−50.66 %)**	**(−0.43 %)**	(−0.90 %)	(−30.43 %)		(−41.35 %)	(−0.76 %)
0.8	0.0107	0.7741	0.0099	0.7760	0.7690	0.0128	0.8	0.0105	0.7820
	(−53.28 %)	(−2.31 %)	(−56.39 %)	(−2.04 %)	(−0.90 %)	(−30.43 %)		(−49.52 %)	(−1.14 %)

Pre-processing methods

Method	**Bias**	**Accuracy**
Massaging	0.0135 (−41.05 %)	0.7740 (−2.32 %)
Reweighting	0.0127 (−44.54 %)	0.7710 (−2.70 %)

Post-processing method

Method	**Bias**	**Accuracy**
ROC	0.0115 (−49.78 %)	0.7690 (−2.95 %)

only three for the logistic model. Adversarial debiasing, as λ_{adv} varied over the same range, produced five bias–accuracy trade-offs marginally inferior to those of the proposed MLP. More importantly, consistent with the training loss behavior, the proposed method achieves a favorable compromise for $\alpha \in [0.62, 0.74]$, where substantial bias reduction is obtained with minimal loss in accuracy.

This trade-off is further quantified in Table 3, which summarizes the numerical performance of all approaches. At $\alpha = 0.7$, both the MLP- and FT-Transformer-based variants of the proposed method halve the original bias while keeping the accuracy loss below 1 %, outperforming the other strategies, none of which achieve comparable bias mitigation without substantially larger drops in accuracy.

5.2.1. Surrogate bound analysis

The theoretical discussion in Section 4 highlights that the proposed surrogate defines a lower bound whose tightness depends on predictive uncertainty, as quantified by the entropy of the class-probability scores $\pi_\theta(\mathbf{x})$. To empirically characterize this effect in the synthetic setting, an additional analysis was carried out focusing on the operating point $\alpha = 0.7$, previously identified as a favorable bias–accuracy trade-off, for the MLP backbone.

The experiment consists of collecting predictions on the test set and computing the following quantities:

- the true MI between the sensitive attributes $\mathbf{z}$ and the hard classifier decisions $g_\theta(\mathbf{x})$, obtained by thresholding the deterministic class-probability scores at $\tau = 0.5$;
- the differentiable surrogate.

To study how the discrepancy between these two quantities depends on predictive uncertainty, the test samples are partitioned into six bins according to the average predictive entropy.

$$\bar{H}_b(\pi_\theta(\mathbf{x})) = -\frac{1}{|S_b|}\sum_{i\in S_b}\Big[\pi_\theta(\mathbf{x}^{(i)})\log\pi_\theta(\mathbf{x}^{(i)}) + \big(1-\pi_\theta(\mathbf{x}^{(i)})\big)\log\big(1-\pi_\theta(\mathbf{x}^{(i)})\big)\Big],$$

where S_b denotes the set of test samples falling into the b-th entropy bin. Bins cover the range of observed entropies and are contiguous but not necessarily equidense, so the number of samples in each bin is determined by the empirical distribution of $\pi_\theta(\mathbf{x})$. For each bin, the average predictive entropy, the true MI, and the surrogate MI are estimated, and their difference is reported as a "gap" curve. The resulting trends are summarized in Fig. 7, which shows that, for the considered operating point, the true MI remains approximately constant across entropy bins, with values close to the test-set bias reported in Table 3 for $\alpha = 0.7$. This behavior is consistent with the fact that the dependency between $\mathbf{z}$ and the hard decisions is largely driven by global structural relations in the synthetic data-generating process rather than by local variations in predictive confidence.

In contrast, the surrogate exhibits a clear decreasing trend as the average predictive entropy increases, consistent with the theoretical analysis of the Bernoulli relaxation. For low-entropy bins, where the classifier is confident (predicted probabilities close to 0 or 1), the surrogate closely matches the true MI and the gap remains very small (typically below 2 % of the local MI). As entropy increases and predictions concentrate around the decision boundary, the surrogate becomes progressively more conservative, leading to a systematically larger gap. In the highest-uncertainty bins, this gap reaches approximately 20–25 % of the corresponding true MI, with intermediate bins remaining well below 10 %. Overall, the deviations remain moderate relative to the bias levels observed in the main synthetic experiments and are comparable to the variations induced by small changes in α in Table 3.

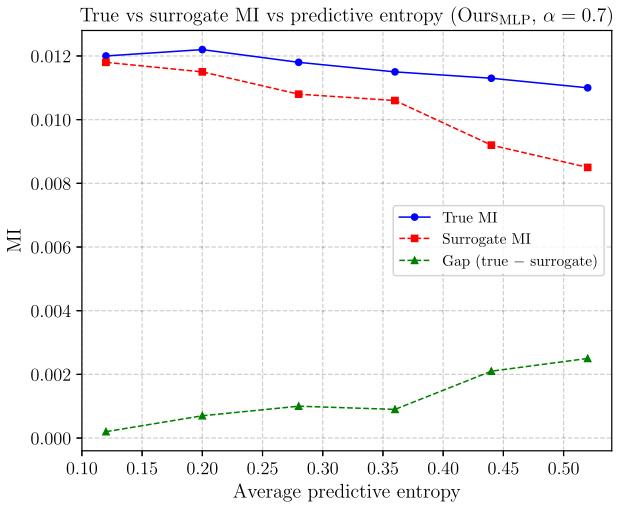


Fig. 7. Comparison between the true MI and the differentiable surrogate as a function of the average predictive entropy, evaluated on the synthetic test set for the proposed approach with an MLP backbone at $\alpha = 0.7$. The true MI remains approximately constant across uncertainty levels, whereas the surrogate exhibits a progressively larger underestimation as entropy increases, reflecting its conservative behavior in high-uncertainty regions. The gap curve quantifies this deviation but remains moderate relative to the corresponding MI values, consistent with the theoretical properties of the surrogate bound.

5.2.2. *Counterfactual fairness*

One of the key advantages of conducting experiments in a synthetic framework is the ability to manipulate the dependency structure among dataset attributes, thanks to full knowledge of the data generation process. This enables the modeling of causal relationships involving sensitive attributes, thereby supporting rigorous counterfactual analysis to evaluate the fairness properties of the proposed in-processing strategy. Counterfactual scenarios are constructed by altering the sensitive attributes $\mathbf{z}$ and updating the associated non-sensitive attributes $\mathbf{x}$ accordingly. This adjustment is performed using the known structural dependencies, whereby new values (x_1^*, x_2^*) are sampled from the conditional distribution $p(x_1, x_2 \mid z_1^*, z_2^*)$, while $(x_3, \ldots, x_{10})$ are held fixed, as they are independent of the sensitive features. Here, the superscript $*$ denotes the counterfactually modified values. Let $\mathbf{x}_{\mathbf{z}}^{*(i)}$ denote the full input vector for sample i, counterfactually generated under a specific assignment $\mathbf{z}$ of sensitive attributes. That is, it includes the modified values for the sensitive-dependent features (x_1, x_2), conditioned on the sensitive assignment $\mathbf{z}$, along with the unchanged values $(x_3, \ldots, x_{10})$. To quantify counterfactual fairness, two complementary metrics are adopted: Counterfactual Mean Absolute Difference (CF-MAD) and Counterfactual Flip Rate (CF-FR). CF-MAD captures the average absolute change in the predicted probabilities due to counterfactual perturbations, accounting for subtle shifts in model confidence even when the final predicted class remains unchanged. It is defined as:

$$\text{CF-MAD} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|\mathcal{P}_{\mathbf{z}}|}\sum_{(\mathbf{z},\mathbf{z}')\in\mathcal{P}_{\mathbf{z}}}\left|p_\theta(y \mid \mathbf{x}_{\mathbf{z}}^{*(i)}) - p_\theta(y \mid \mathbf{x}_{\mathbf{z}'}^{*(i)})\right|, \tag{51}$$

where $\mathcal{P}_{\mathbf{z}}$ denotes the set of all distinct pairs of sensitive attribute values, and $|\mathcal{P}_{\mathbf{z}}|$ is the number of such pairs.

Conversely, CF-FR measures the proportion of instances for which counterfactual changes in sensitive attributes result in different binary classification decisions, providing an interpretable measure of decision-level instability. CF-FR is defined as:

$$\begin{aligned}\text{CF-FR} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{|\mathcal{P}_{\mathbf{z}}|}\sum_{(\mathbf{z},\mathbf{z}')\in\mathcal{P}_{\mathbf{z}}}\mathbf{1}\big\{\mathbf{1}\{p_\theta(y \mid \mathbf{x}_{\mathbf{z}}^{*(i)}) \geq 0.5\} \\ \neq \mathbf{1}\{p_\theta(y \mid \mathbf{x}_{\mathbf{z}'}^{*(i)}) \geq 0.5\}\big\}.\end{aligned} \tag{52}$$

The results of the counterfactual fairness analysis are summarized in Fig. 8 and Table 4, which report both CF-MAD and CF-FR as functions of the regularization parameter α. The CF-MAD curve exhibits a smooth, gradual decline for $\alpha \lesssim 0.6$, followed by a steeper drop beyond this threshold. This behavior is expected, as CF-MAD is sensitive to continuous changes in predicted probabilities. In contrast, CF-FR remains relatively flat across smaller values of α, but undergoes a sharp decline once α exceeds 0.6. This abrupt drop reflects the nature of CF-FR, which depends on discrete changes around the classification threshold (typically 0.5) and may be influenced by practical decision rules. Both nonlinear model variants are shown: the FT-Transformer follows the same qualitative pattern as the MLP but attains lower CF-MAD and CF-FR at comparable α—especially beyond $\alpha \approx 0.6$—indicating a more favorable counterfactual-fairness profile on this synthetic dataset.

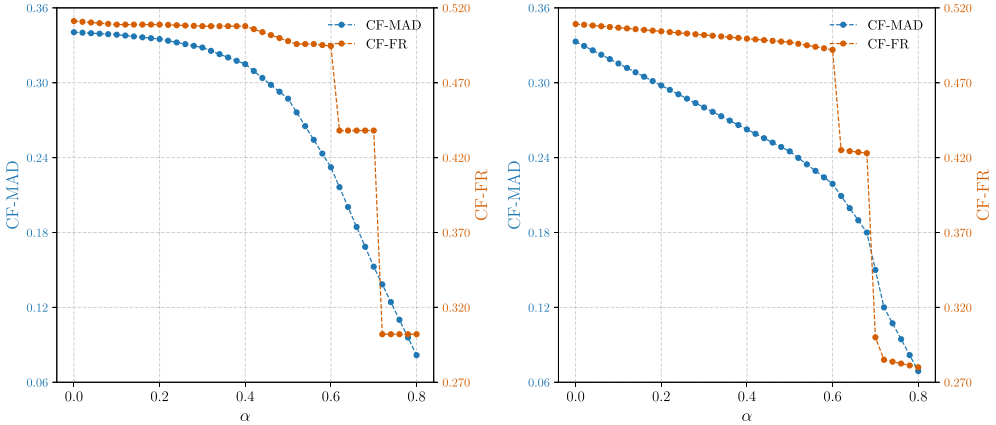


Fig. 8. Counterfactual fairness analysis showing CF-MAD (left y-axis, blue) and CF-FR (right y-axis, orange) as functions of the regularization parameter α, evaluated on the test set, for the two model variants: MLP (left panel) and FT-Transformer (right panel). In both panels, CF-MAD decreases gradually up to $\alpha \approx 0.5$ and then drops sharply; CF-FR is nearly flat at low α and decreases abruptly beyond $\alpha \approx 0.6$. Relative to the MLP, the FT-Transformer attains lower CF-MAD and CF-FR at comparable α—especially past the knee—indicating a more favorable counterfactual-fairness profile on this dataset.

Table 4
Test set results for CF-MAD and CF-FR across different α values. Percentage variations are reported relative to the baseline case ($\alpha = 0$).

α	Ours_{MLP}		Ours_{FT}	
	CF-MAD	**CF-FR**	**CF-MAD**	**CF-FR**
0.0	0.3405	0.5112	0.3330	0.5092
0.1	0.3385	0.5088	0.3154	0.5068
	(−0.59 %)	(−0.47 %)	(−5.29 %)	(−0.47 %)
0.2	0.3350	0.5088	0.2978	0.5043
	(−1.62 %)	(−0.47 %)	(−10.57 %)	(−0.96 %)
0.3	0.3282	0.5078	0.2802	0.5019
	(−3.61 %)	(−0.67 %)	(−15.86 %)	(−1.43 %)
0.4	0.3149	0.5078	0.2626	0.4994
	(−7.52 %)	(−0.67 %)	(−21.14 %)	(−1.92 %)
0.5	0.2872	0.4978	0.2450	0.4970
	(−15.65 %)	(−2.62 %)	(−26.43 %)	(−2.40 %)
0.6	0.2323	0.4945	0.2190	0.4920
	(−31.78 %)	(−3.27 %)	(−34.23 %)	(−3.38 %)
0.7	**0.1526**	**0.4381**	**0.1500**	**0.3000**
	(−55.18 %)	**(−14.30 %)**	**(−54.95 %)**	**(−41.08 %)**
0.8	0.0817	0.3021	0.0690	0.2800
	(−76.01 %)	(−40.90 %)	(−79.28 %)	(−45.01 %)

Table 5
Execution time comparison between the proposed gradient-based method and a gradient-free GA, across increasing dataset sizes.

Dataset size	Ours_{MLP}	GA
5000	12 min 16 s	34 min 0 s
10000	17 min 27 s	3 h 5 min 25 s
50000	47 min 20 s	14 h 49 min 20 s
100000	1 h 35 min 47 s	1 d 5 h 10 min 0 s
500000	11 h 48 min 32 s	18 d 14 h 9 min 0 s
1000000	18 h 40 min 25 s	38 d 15 h 5 min 30 s *(estimated)*

These results are consistent with earlier findings from standard performance evaluations, reaffirming that the range $0.6 \lesssim \alpha \lesssim 0.7$ corresponds to an optimal trade-off zone where indirect bias is substantially mitigated with minimal loss in accuracy. The consistency of trends across MLP and FT-Transformer further supports the robustness of the proposed regularization with respect to the underlying nonlinear architecture. Furthermore, the pronounced reduction in counterfactual unfairness metrics observed beyond this threshold underscores the effectiveness of the proposed fairness regularization in addressing indirect discrimination.

5.2.3. Computational complexity

Another key advantage of using a synthetic framework lies in the ability to generate datasets of arbitrary size, enabling systematic investigations into algorithmic complexity and scalability. Since one of the principal benefits of the proposed approach is the ability to employ gradient-based optimization—as opposed to more computationally demanding gradient-free methods—a comparative runtime analysis was conducted against GA, representing a popular gradient-free technique.

The execution time of both methods was evaluated across six configurations with increasing sample sizes N, and the results are summarized in Table 5. For the GA, the population size was fixed at 200, and the algorithm was run for 150 generations. A single-point crossover was used with a probability of 0.8, per-parameter mutation was applied with a probability of 0.02 (using Gaussian noise with a standard deviation of 0.1), and tournament selection of size 3 was employed. These hyperparameter choices aim to strike a balance between solution diversity and computational tractability across different dataset scales. Increasing the population size or number of generations further would significantly inflate the execution time, especially for large datasets, whereas reducing them would risk insufficient exploration. Similarly, the chosen mutation and selection strategies are designed to prevent premature convergence while maintaining effective evolutionary pressure. For analogous reasons, and to ensure a fair comparison, the MLP-based architecture was used in this analysis.

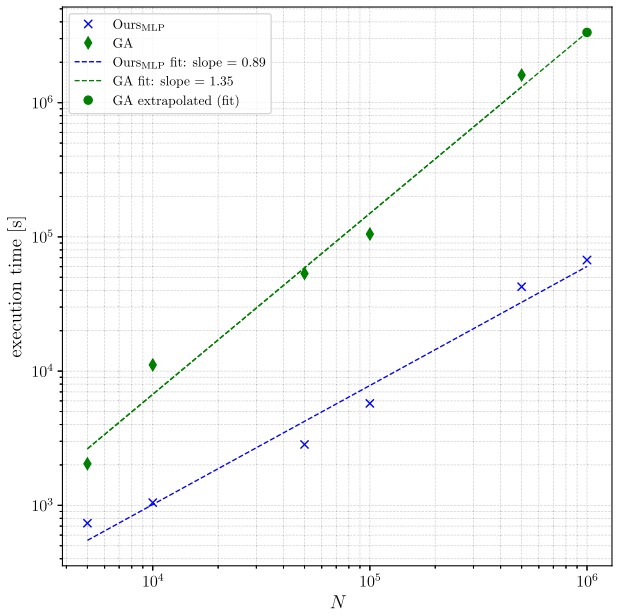


Fig. 9. Computational complexity analysis comparing the proposed gradient-based method and a gradient-free GA, in terms of execution time across increasing synthetic dataset sizes. Both axes are shown on a log–log scale. Each point corresponds to the training time for a given number of samples, with dashed lines indicating linear fits on the log-transformed data.

Crucially, the same GA configuration was applied across all dataset sizes in order to isolate the effect of scaling with N, without conflating results with variations in algorithmic settings.

Both methods were executed under the same computational conditions and took advantage of parallelism wherever appropriate. In the proposed method, MC samples and training batches were processed jointly to reduce overhead, and parallel computation was used to accelerate both forward and backward passes. For the GA, population members were evaluated in parallel batches to the extent allowed by available memory. However, due to the structure of evolutionary algorithms, individual models differ across the population and must be processed independently during selection, crossover, and mutation. These operations involve additional sequential steps and repeated access to intermediate representations, which inherently limit parallelism and increase overhead. While both methods benefited from parallel execution, these structural differences contributed to the observed discrepancy in scalability.

The results of this comparison are shown in Fig. 9, where execution time is plotted against dataset size N on a log-log scale. A linear relationship in this representation indicates a power-law dependence of the form:

$$\log(\text{time}) = \delta \log(N) + \text{constant} \quad \rightarrow \quad \text{time} \sim N^{\delta}. \tag{53}$$

In this formulation, a slope $\delta > 1$ corresponds to superlinear growth, while $\delta < 1$ indicates sublinear behavior. Empirically, the GA exhibits a slope of approximately 1.35, indicating superlinear scaling. In contrast,

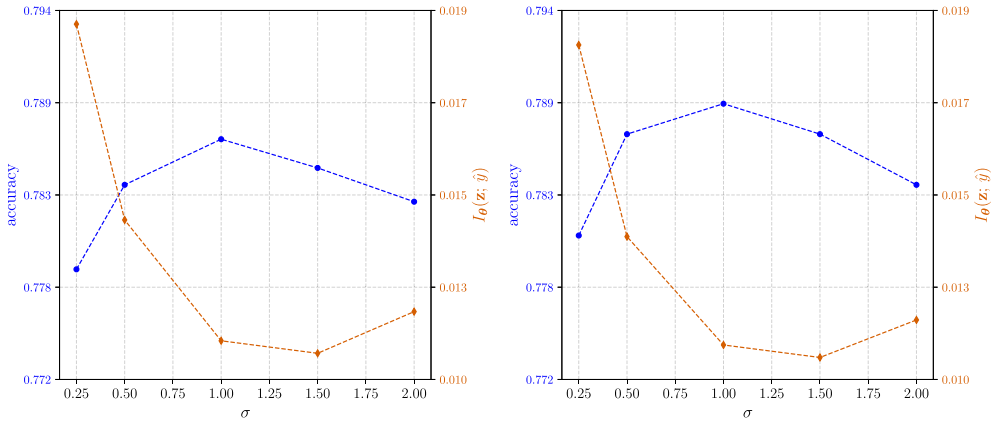


Fig. 10. Sensitivity of the proposed pipeline to kernel bandwidth σ at $\alpha = 0.7$. The left and the right panels show the MLP-based and FT-Transformer–based variants, respectively. Very small σ produce under-smoothed pseudo-counts and increased bias with a slight accuracy drop; $\sigma \in [1, 1.5]$ stabilizes the surrogate and reduces bias with negligible impact on accuracy; excessively large σ (e.g., 2.0) mildly degrades both metrics due to over-smoothing.

the proposed approach shows a significantly lower slope of about 0.89, revealing sublinear scaling with respect to dataset size.

The discrepancy between the theoretical linear bound reported in Section 4.4 and the observed $\delta < 1$ arises because, in practice, the effective epoch count tends to saturate or slightly decrease for large N due to early stopping, and because GPU-level parallelization amortizes the cost of MC passes.

Although the full training pipeline includes both kernel smoothing and stochastic regularization via dropout sampling—each introducing additional overhead—their contributions remain well-behaved in practice. Kernel smoothing is performed once offline with a fixed number of nearest neighbors and contributes a modest, near-linear preprocessing cost that is negligible compared to the total training time. Meanwhile, MC dropout introduces a multiplicative factor on the per-batch training cost proportional to the number of samples, but this cost is efficiently distributed across parallel operations. In practice, this results in runtime multipliers that remain below the theoretical upper bound derived in Section 4.4.

These findings demonstrate not only that the proposed method requires lower absolute execution time for large datasets, but also that it offers superior computational scalability relative to the GA baseline. For $N = 1{,}000{,}000$, the GA's runtime becomes prohibitively long and was extrapolated from the fitted power-law model for practical purposes.

5.2.4. Sensitivity analysis

In this subsection a sensitivity analysis is conducted on three hyperparameters central to the proposed pipeline—kernel bandwidth σ, number of MC samples T, and dropout probability p_{drop}—using the two most effective backbones (MLP and FT-Transformer). To isolate the effect of these hyperparameters from the fairness–utility trade-off, the analysis fixes the regularization weight to $\alpha = 0.7$, which was previously identified as the best operating point on the test set for both backbones; keeping α constant in this regime avoids confounding changes in the bias–accuracy frontier and yields a representative high-performance baseline. For each study, a single hyperparameter is varied while all others are fixed to the synthetic defaults ($\sigma = 1$, $k = 8$, $T = 10$, $p_{\text{drop}} = 0.2$), and performance is reported on the held-out test set. The explored ranges are selected to cover the practically relevant regimes around the validated defaults while exposing under- and over-regularized behaviors: $\sigma \in \{0.25, 0.50, 1.00, 1.50, 2.00\}$ spans from an under-smoothed regime—where pseudo-counts are sparse and the MI surrogate is noisy—through the validated neighborhood around $\sigma = 1$ up to an over-smoothed regime that can blur informative local structure; $T \in \{1, 2, 5, 10, 20, 40\}$ moves from minimal averaging—higher estimator variance and noisier gradients—toward larger averages where diminishing returns are expected, with choices that remain computationally affordable given the linear cost in T; $p_{\text{drop}} \in \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ ranges from no stochasticity to moderately strong dropout, excluding extreme values that typically harm capacity without providing further fairness benefits. Results are summarized in Figs. 10–12; each figure contains two panels (MLP on the left and FT-Transformer on the right) and overlays test accuracy on the left y-axis with the bias on the right y-axis. Fig. 10 shows that very small bandwidths (e.g., $\sigma = 0.25$) yield under-smoothed pseudo-counts, which manifest as higher bias and a slight reduction in accuracy; moving toward $\sigma \in [1, 1.5]$ stabilizes the surrogate and reduces bias with negligible impact on accuracy, whereas excessively large values (e.g., $\sigma = 2.0$) mildly degrade both metrics due to over-smoothing. Fig. 11 indicates that increasing T monotonically decreases the bias with diminishing returns beyond $T \approx 10$, consistent with variance reduction in the surrogate estimate and a more reliable optimization signal; accuracy remains essentially stable, with variations limited to a few tenths of a percent, suggesting that improvements stem from lower estimator noise rather than from changes in effective regularization strength. Fig. 12 highlights that moving from no dropout to moderate levels (0.2–0.35) lowers bias at a modest accuracy cost, in line with the conservative behavior of the uncertainty-aware MI under stochastic subnet sampling; pushing p_{drop} to high values (≥ 0.5) begins to impair model capacity and no longer improves bias. Taken together, the three figures delineate robust operating regions shared across backbones—$\sigma \in [1, 1.5]$, $T \in [10, 20]$, and $p_{\text{drop}} \in [0.2, 0.35]$—that attain near-minimal bias with negligible or limited changes in accuracy, whereas extreme settings either under-capture dependence or induce over-regularization.

5.3. Real dataset experiments

The same experimental procedure adopted for the synthetic framework was applied to the 2001 Dutch Census dataset, and the results exhibit a consistent pattern. In particular, Fig. 13, which illustrates the bias–accuracy trade-off on the test set as a function of the regularization parameter α, confirms that improvements in fairness (i.e., reductions in bias) are generally accompanied by a decrease in classification performance. However, this trade-off does not occur uniformly; the magnitude of accuracy loss is significantly smaller than the corresponding bias reduction in several regions of α. Notably, even in this non-idealized, real-world scenario, it is possible to identify values of α that yield

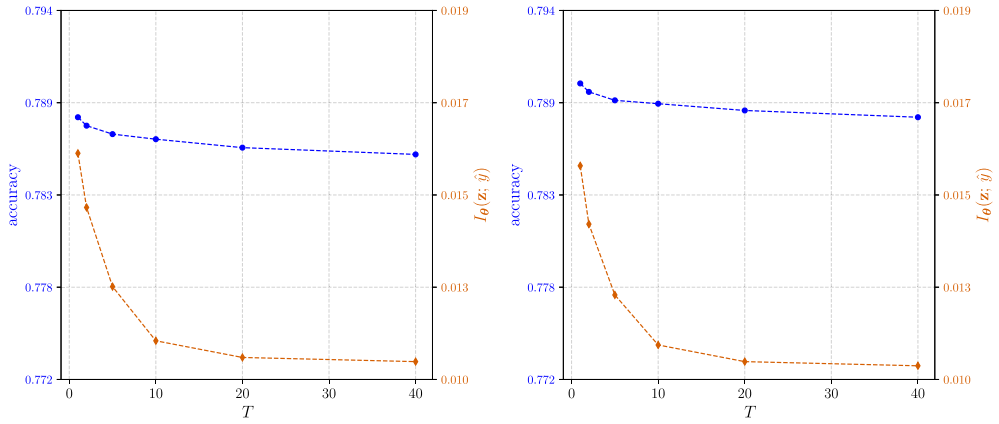


Fig. 11. Sensitivity of the proposed pipeline to the number of MC samples T at $\alpha = 0.7$. The left and the right panels show the MLP-based and FT-Transformer–based variants, respectively. Increasing T reduces bias with diminishing returns beyond $T \approx 10$, while accuracy remains essentially stable (changes within a few tenths of a percent); very small T (e.g., $T = 1$) yields higher estimator variance and slightly noisier optimization, inflating bias, whereas very large T (e.g., $T = 40$) provides marginal additional benefits at increased cost.

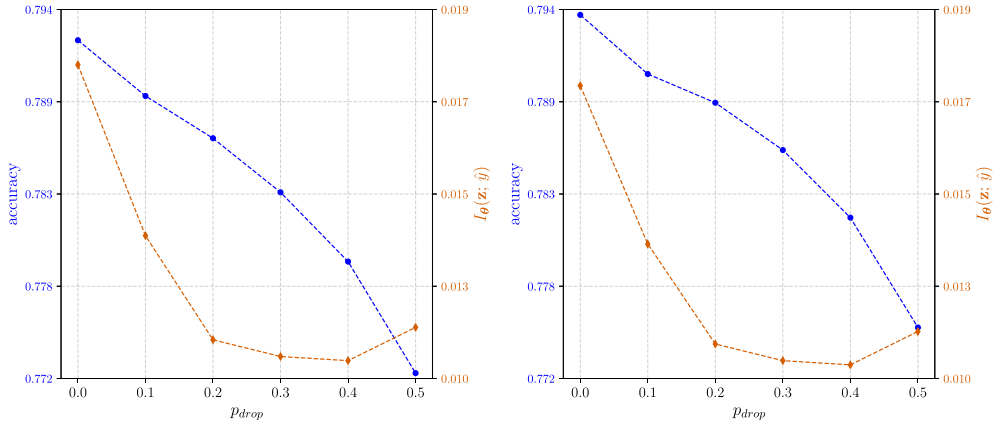


Fig. 12. Sensitivity of the proposed pipeline to dropout probability p_{drop} at $\alpha = 0.7$. The left and the right panels show the MLP-based and FT-Transformer–based variants, respectively. Moving from no dropout to moderate levels (0.2–0.35) reduces bias at a modest accuracy cost; very high p_{drop} (≥ 0.5) impairs capacity and no longer improves bias, while $p_{\text{drop}} = 0$ under-captures uncertainty and yields higher bias.

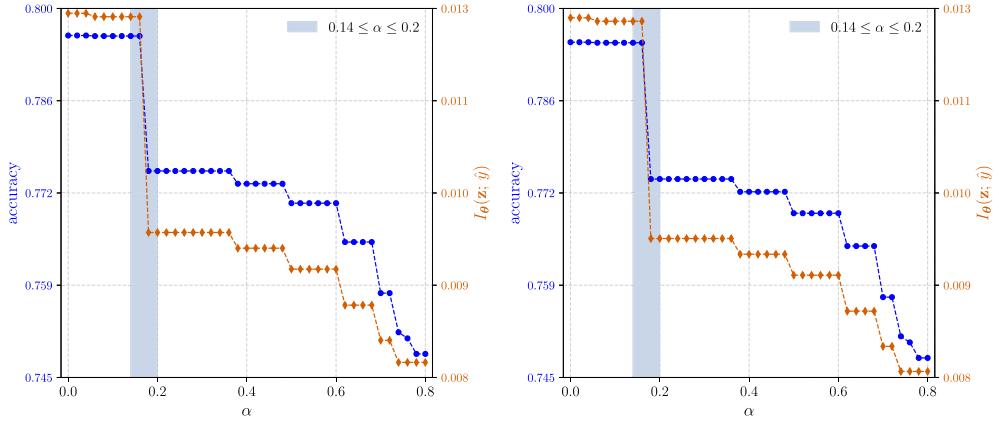


Fig. 13. Bias–accuracy trade-off on the test set of the 2001 Dutch Census dataset as a function of α for two backbones: MLP (left panel) and FT-Transformer (right panel). Accuracy (left y-axis, blue) and bias measured via MI (right y-axis, orange) both decrease in a step-wise manner as α increases. The shaded band highlights $\alpha \in [0.14, 0.20]$, where a marked bias reduction occurs with only a modest loss in accuracy.

favorable trade-offs. In the examined case, setting $\alpha \approx 0.2$ yields a substantial reduction in model bias—approximately 23 %—with only a ≈ 2.5 % decrease in accuracy. This behavior confirms the effectiveness of the fairness regularization mechanism under real-world conditions, where the data are not artificially constructed but represent genuine societal and economic patterns.

Table 6
Comparison of test-time bias and accuracy on the 2001 Dutch Census data for the proposed approach, in all its variants, and the adversarial debiasing method, evaluated across varying values of the regularization parameter α (λ_{adv} for the adversarial debiasing). Results from two pre-processing baselines—massaging and reweighting—and the ROC post-processing approach are included as well. For each method, percentage changes are reported in parentheses relative to the same method with no bias mitigation ($\alpha = 0$). For the pre-processing and post-processing strategies, the percentage changes are relative to the proposed approach under the MLP backbone and $\alpha = 0$, as they share the same model architecture.

α	Ours_{MLP}		Ours_{FT}		Ours_{Log}		λ_{adv}	Adv. Debiasing	
	Bias	**Accuracy**	**Bias**	**Accuracy**	**Bias**	**Accuracy**		**Bias**	**Accuracy**
0.0	0.0126	0.7960	0.0126	0.7950	0.0112	0.7764	0.0	0.0126	0.7960
0.1	0.0126	0.7959	0.0125	0.7949	0.0112	0.7764	0.1	0.0121	0.7928
	(0.00 %)	(−0.01 %)	(−0.79 %)	(−0.01 %)	(0.00 %)	(0.00 %)		(−4.12 %)	(−0.40 %)
0.2	**0.0097**	**0.7758**	**0.0096**	**0.7746**	0.0112	0.7764	0.2	0.0113	0.7880
	(**−23.02** %)	(**−2.54** %)	(**−23.81** %)	(**−2.57** %)	(0.00 %)	(0.00 %)		(−10.86 %)	(−1.01 %)
0.3	0.0097	0.7758	0.0096	0.7746	0.0112	0.7764	0.3	0.0104	0.7805
	(−23.02 %)	(−2.54 %)	(−23.81 %)	(−2.57 %)	(0.00 %)	(0.00 %)		(−17.59 %)	(−1.95 %)
0.4	0.0095	0.7739	0.0094	0.7727	0.0112	0.7764	0.4	0.0104	0.7805
	(−24.60 %)	(−2.78 %)	(−25.40 %)	(−2.81 %)	(0.00 %)	(0.00 %)		(−17.59 %)	(−1.95 %)
0.5	0.0092	0.7710	0.0092	0.7695	0.0111	0.7763	0.5	0.0104	0.7805
	(−26.98 %)	(−3.14 %)	(−26.98 %)	(−3.21 %)	(−0.89 %)	(−0.01 %)		(−17.59 %)	(−1.95 %)
0.6	0.0092	0.7710	0.0092	0.7695	0.0111	0.7761	0.6	0.0099	0.7710
	(−26.98 %)	(−3.14 %)	(−26.98 %)	(−3.21 %)	(−0.89 %)	(−0.04 %)		(−21.95 %)	(−3.14 %)
0.7	0.0083	0.7576	0.0082	0.7570	0.0106	0.7667	0.7	0.0099	0.7710
	(−34.13 %)	(−4.82 %)	(−34.92 %)	(−4.78 %)	(−5.36 %)	(−1.25 %)		(−21.95 %)	(−3.14 %)
0.8	0.0080	0.7485	0.0079	0.7479	0.0101	0.7619	0.8	0.0096	0.7650
	(−36.51 %)	(−5.97 %)	(−37.30 %)	(−5.92 %)	(−9.82 %)	(−1.87 %)		(−23.93 %)	(−3.89 %)

Pre-processing methods

Method	**Bias**	**Accuracy**
Massaging	0.0099 (−21.51 %)	0.7631 (−4.13 %)
Reweighting	0.0097 (−23.25 %)	0.7613 (−4.36 %)

Post-processing method

Method	**Bias**	**Accuracy**
ROC	0.0096 (−23.81 %)	0.7590 (−4.65 %)

On the Dutch Census, the FT-Transformer– and MLP-based variants exhibit comparable performance. Over $\alpha \in [0.18, 0.36]$, both attain advantageous bias–accuracy trade-offs; this is consistent with the dataset's lower effective complexity, which limits the incremental benefit of architectures more expressive than an MLP. Relative to the baseline mitigations (massaging, reweighting, adversarial debiasing), both variants trace a more favorable trade-off frontier and, compared with the linear logistic counterpart, offer greater flexibility by yielding a denser set of operating points along the bias–accuracy curve. Quantitative results are summarized in Table 6 and visualized in Fig. 14, further motivating the adoption of the proposed in-processing pipeline within nonlinear architectures.

6. Limitations and future directions

While the proposed framework demonstrates strong empirical performance, several limitations remain. These span practical, methodological, and theoretical dimensions, and offer promising avenues for future research.

The first limitation concerns the assumption that the input space $\mathcal{X}$ is discrete. This design choice facilitates the estimation of joint and marginal distributions via kernel-weighted counts and aligns with many benchmark datasets in the fairness literature—such as the 2001 Dutch Census. However, real-world data often include continuous or mixed-type features. While discretization is a common workaround, it introduces sensitivity to binning schemes and results in information loss. A more principled approach could involve an adaptation of the proposed surrogate to variational MI estimators, such as MINE [33] or CLUB [34], which support continuous inputs through learned discriminators.

Another limitation concerns the scalability of the kernel smoothing strategy in high-dimensional input spaces. While the proposed pipeline is explicitly designed for compatibility with nonlinear and high-dimensional data, it can become computationally burdensome when dimensionality reaches extreme levels. Kernel truncation (via k-nearest neighbors) partially addresses this issue, but it does not fully overcome the curse of dimensionality, which degrades the informativeness of Hamming distances and can lead to sparse pseudo-counts. Scalable alternatives include locality-sensitive hashing, product kernels exploiting feature independence, or factorized models that approximate MI component-wise [47].

The choice of the kernel itself is nontrivial and can significantly influence the quality and stability of the smoothed estimates. Standard kernels like exponential or triangular forms may not generalize well across heterogeneous feature spaces or mixed-type variables. Automatic kernel selection strategies—such as data-driven metric learning [48] or multiple kernel learning [49]—have shown promise in other domains and could enhance adaptivity in fairness-aware settings, particularly when group-wise data characteristics vary.

The use of MC dropout in the proposed method serves as a lightweight Bayesian approximation that mitigates model overconfidence and enhances the reliability of the fairness surrogate. The current implementation exploits simple averaging across dropout-induced subnetworks. While this is standard and yields low-variance estimates [50], future work may consider robust aggregation strategies, such as trimmed means or entropy-weighted averaging [51], which can reduce the influence of subnetworks that produce unstable predictions due to parameter masking. Any such strategy must retain differentiability to preserve gradient flow through the fairness surrogate. Extending uncertainty modeling to the utility loss may also yield more coherent training dynamics, provided that variance reduction techniques are used to ensure stable gradients.

Robustness across domains is another key consideration. A single set of hyperparameters was applied to both synthetic and real-world datasets in this work, showing promising generalization. Still, a fixed grid search limits exploration and may overlook better fairness–utility trade-offs. This issue is heightened in dynamic or non-stationary

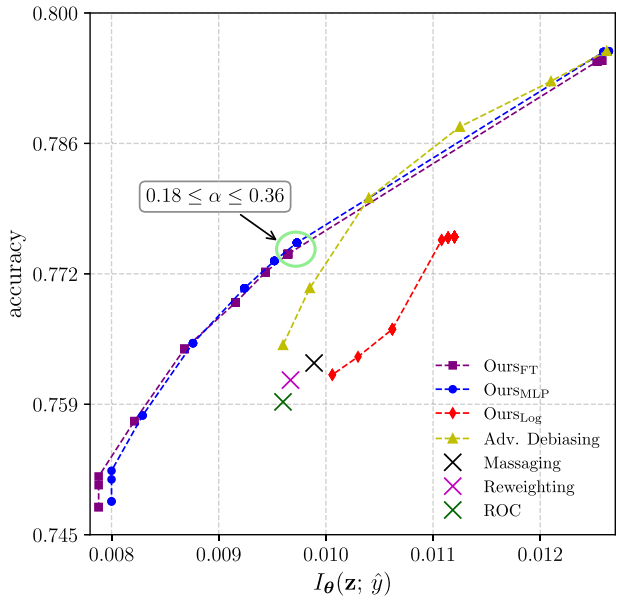


Fig. 14. Bias–accuracy trade-off on the test set of the 2001 Dutch Census dataset for the proposed in-processing method with three backbones—MLP and FT-Transformer (nonlinear) and logistic (linear)—plus an in-processing adversarial debiasing strategy, two pre-processing baselines (massaging, reweighting) and a post-processing approach (ROC). Each point represents test performance for a given value of the regularization parameter α; for adversarial debiasing, each point corresponds to a value of the adversarial scaling coefficient λ_{adv}; overlapping points indicate repeated results for different regularization coefficient values. In its nonlinear variants, the proposed method traces a consistently stronger bias–accuracy frontier than the alternative approaches. The green-circled region highlights α values in the range $[0.18,\ 0.36]$, where a favorable balance between bias mitigation and accuracy emerges.

environments. Adaptive tuning, meta-learning-based strategies [52], or domain-adaptive surrogates could improve flexibility and robustness [53]. In this regard, results from the sensitivity analysis—characterized by broad plateaus and stable trends—suggest that hyperparameter self-adaptation could replace static grids with slow, data-driven adjustments. A particularly promising target is the trade-off weight α, whose conservative, feedback-guided updates based on held-out bias measurements may stabilize the fairness–utility balance under distributional drift; complementary, cautious schedules for σ, T, and p_{drop} keyed to estimator stability could track the observed "sweet spots" without full re-tuning. Related ideas have proved effective in adjacent areas through diversified or fine-grained adaptive regularization and nonlinear canonical polyadic decompositions for high-dimensional, incomplete data [54–56], indicating a feasible path to integrate analogous self-adaptation mechanisms into the present framework as a future direction.

Nevertheless, the sensitivity analysis conducted in Section 5 revealed wide stability plateaus for both σ and p_{drop}, indicating that the proposed pipeline remains robust across broad ranges of these hyperparameters. This empirical evidence supports the use of fixed settings in the present study, avoiding unnecessary online overhead while preserving the sub-linear runtime growth with dataset size established in Section 4.4. In more complex or dynamically evolving scenarios, however, adaptive strategies could further enhance flexibility and fairness generalization—for instance, σ could be adjusted online based on local data density (e.g., via plug-in estimators or median-heuristic updates), while p_{drop} could be modulated as a function of epistemic uncertainty. In such cases, particular care should be devoted to mitigating the computational overhead introduced by per-batch updates so as to maintain the scalability advantages of the proposed approach. These considerations open a broad avenue for future research on adaptive yet efficient extensions of the framework.

Lastly, it is worth noting that the fairness objective employed in this work targets demographic parity, expressed as independence between predicted labels and sensitive attributes. While this is a general and powerful statistical fairness criterion, it does not inherently guarantee properties like equalized odds or equal opportunity. In applications where fairness policies are more nuanced, the scalar trade-off parameter α may lack sufficient expressiveness to balance competing objectives. Future extensions could explore multi-objective optimization [57], fairness-constrained learning formulations [58], or dynamic fairness weighting based on policy-specific requirements [59].

7. Conclusion

This study proposes a fairness-aware training strategy that embeds a differentiable, uncertainty-aware surrogate of mutual information within standard neural classifiers. By combining Bernoulli relaxation, kernel smoothing, and Monte Carlo dropout, the resulting regularizer integrates seamlessly into gradient-based optimization and remains model-agnostic, promoting statistical independence between predictions and sensitive attributes while preserving competitive utility. Empirical evidence on synthetic and real-world data, together with a complexity analysis, indicates favorable fairness–utility trade-offs with modest computational overhead, making the approach practical for large-scale settings. Looking ahead, extending the objective beyond demographic parity, exploring adaptive or learned kernels and temperature schedules, and developing multi-objective or constraint-based formulations tailored to application-specific policies represent promising directions for further refinement and deployment.

CRediT authorship contribution statement

Alessandro Incremona: Writing – original draft, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Andrea Pozzi:** Writing – review & editing, Writing – original draft, Methodology, Investigation, Conceptualization. **Alessandro Guiscardi:** Writing – review & editing, Investigation. **Daniele Tessera:** Writing – review & editing, Supervision, Project administration, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research was partially funded by the European Union through the European Social Fund (FSE) under the Recovery Assistance for Cohesion and the Territories of Europe (REACT-EU) initiative, within the context of the National Operational Programme (PON) on Research and Innovation 2014–2020, pursuant to DM 1062/2021, Contract No. 57-I-999-6. The authors express their gratitude for the support provided.

Appendix A. Implementation hyperparameters summary

This appendix summarizes all implementation hyperparameters adopted in the proposed framework across its three model variants—Logistic Regression, Multi-Layer Perceptron (MLP), and FT-Transformer—to improve clarity and facilitate reproducibility. The regularization coefficient α is not included here, as its influence on fairness–utility trade-offs was systematically analyzed in Section 5.

Table A.1
Implementation hyperparameters for the proposed method.

Hyperparameter	Synthetic framework	Real-world framework
Logistic Regression — penalty	$\ell_2 = 0.1$ (ridge)	$\ell_2 = 0.1$ (ridge)
MLP — hidden layers	[64, 32] units, ReLU activations	[32, 32] units, ReLU activations
FT-Transformer — token dimension d_{token}	64	64
FT-Transformer — encoder blocks N_{blocks}	3	3
FT-Transformer — attention heads N_{heads}	4	4
FT-Transformer — feed-forward hidden size d_{ff}	128 (expansion factor 2)	128 (expansion factor 2)
FT-Transformer — activations	GELU, pre-norm LayerNorm	GELU, pre-norm LayerNorm
FT-Transformer — attention / residual dropout	0.10	0.20
FT-Transformer — token (embedding) dropout	0.05	0.05
FT-Transformer — stochastic depth	0.05	0.05
Kernel function	Exponential	Exponential
Kernel bandwidth σ	1.0	1.0
Truncation criterion	k-nearest neighbors, $k = 8$	k-nearest neighbors, $k = 8$
Dropout rate p_{drop}	0.20	0.20
Monte Carlo samples T	10	10
Learning rate η	0.05	0.05
Optimizer	Stochastic Gradient Descent (SGD)	Stochastic Gradient Descent (SGD)
Batch size B	256	256
Maximum epochs $E_{\max}$	200	200
Early stopping	Patience: 40 epochs; tolerance: 1 % improvement	Patience: 40 epochs; tolerance: 1 % improvement
Utility loss L_u	Binary Cross-Entropy (BCE)	Binary Cross-Entropy (BCE)
Fairness loss L_f	Differentiable MI surrogate	Differentiable MI surrogate

Data availability

Data will be made available on request.

References

[1] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, A. Galstyan, A survey on bias and fairness in machine learning, ACM Comput. Surv. 54 (6) (2021) 1–35, https://doi.org/10.1145/3457607

[2] E. Barbierato, A. Gatti, A. Incremona, A. Pozzi, D. Toti, Breaking away from ai: The ontological and ethical evolution of machine learning, IEEE Access (2025) https://doi.org/10.1109/ACCESS.2025.3553032

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

[4] R. Berk, H. Heidari, S. Jabbari, M. Kearns, A. Roth, Fairness in criminal justice risk assessments: The state of the art, Soc. Methods Res. 50 (1) (2021) 3–44.

[5] K. Holstein, J. Wortman Vaughan, H. Daumé III, M. Dudik, H. Wallach, Improving fairness in machine learning systems: What do industry practitioners need?, in: Proceedings of the 2019 CHI conference on human factors in computing systems, 2019, pp. 1–16.

[6] S. Dehdashtian, B. Sadeghi, V. N. Boddeti, Utility-fairness trade-offs and how to find them, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 12037–12046.

[7] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, R. Zemel, Fairness through awareness, in: Proceedings of the 3rd innovations in theoretical computer science conference, 2012, pp. 214–226.

[8] S. Verma, J. Rubin, Fairness definitions explained, in: Proceedings of the international workshop on software fairness, 2018, pp. 1–7.

[9] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, S. Venkatasubramanian, Certifying and removing disparate impact, in: proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, 2015, pp. 259–268.

[10] A. Romei, S. Ruggieri, A multidisciplinary survey on discrimination analysis, Knowl. Eng. Rev. 29 (5) (2014) 582–638.

[11] F. Kamiran, T. Calders, Data preprocessing techniques for classification without discrimination, Knowl. Inf. Syst. 33 (1) (2012) 1–33.

[12] P. K. Lohia, K. N. Ramamurthy, M. Bhide, D. Saha, K. R. Varshney, R. Puri, Bias mitigation post-processing for individual and group fairness, in: Icassp 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp), IEEE, 2019, pp. 2847–2851.

[13] T. Kamishima, S. Akaho, H. Asoh, J. Sakuma, Fairness-aware classifier with prejudice remover regularizer, in: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II 23, Springer, 2012, pp. 35–50.

[14] J. Cho, G. Hwang, C. Suh, A fair classifier using mutual information, in: 2020 IEEE international symposium on information theory (ISIT), IEEE, 2020, pp. 2521–2526.

[15] C. Zhao, L. Wu, P. Shao, K. Zhang, R. Hong, M. Wang, Fair representation learning for recommendation: A mutual information perspective, in: Proceedings of the AAAI Conference on artificial intelligence, vol. 37, 2023, pp. 4911–4919.

[16] A. Z. Jacobs, H. Wallach, Measurement and fairness, in: Proceedings of the 2021 ACM conference on fairness, accountability, and transparency, 2021, pp. 375–385.

[17] F. Pennerath, P. Mandros, J. Vreeken, Discovering approximate functional dependencies using smoothed mutual information, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 1254–1264.

[18] U. F. Siddiqi, S. M. Sait, O. Kaynak, Genetic algorithm for the mutual information-based feature selection in univariate time series data, IEEE Access 8 (2020) 9597–9609.

[19] P. Ganesh, H. Chang, M. Strobel, R. Shokri, On the impact of machine learning randomness on group fairness, in: Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency, 2023, pp. 1789–1800.

[20] S. Kuzucu, J. Cheong, H. Gunes, S. Kalkan, Uncertainty as a fairness measure, J. Artif. Intell. Res. 81 (2024) 307–335.

[21] M. B. Zafar, I. Valera, M. G. Rogriguez, K. P. Gummadi, Fairness constraints: Mechanisms for fair classification, in: Artificial intelligence and statistics, PMLR, 2017, pp. 962–970.

[22] A. Cotter, M. Gupta, H. Jiang, N. Srebro, K. Sridharan, S. Wang, B. Woodworth, S. You, Training well-generalizing classifiers for fairness metrics and other data-dependent constraints, in: International Conference on Machine Learning, PMLR, 2019, pp. 1397–1405.

[23] A. Beutel, J. Chen, T. Doshi, H. Qian, A. Woodruff, C. Luu, P. Kreitmann, J. Bischof, E. H. Chi, Putting fairness principles into practice: Challenges, metrics, and improvements, in: Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, 2019, pp. 453–459.

[24] R. Jiang, A. Pacchiano, T. Stepleton, H. Jiang, S. Chiappa, Wasserstein fair classification, in: Uncertainty in artificial intelligence, PMLR, 2020, pp. 862–872.

[25] H. Xu, X. Liu, Y. Li, A. Jain, J. Tang, To be robust or to be fair: Towards fairness in adversarial training, in: International conference on machine learning, PMLR, 2021, pp. 11492–11501.

[26] D. Xu, S. Yuan, L. Zhang, X. Wu, Fairgan: Fairness-aware generative adversarial networks, in: 2018 IEEE international conference on big data (big data), IEEE, 2018, pp. 570–575.

[27] Z. Li, A. Pérez-Suay, G. Camps-Valls, D. Sejdinovic, Kernel dependence regularizers and gaussian processes with applications to algorithmic fairness, Pattern Recognit. 132 (2022) 108922.

[28] N. Deka, D. J. Sutherland, Mmd-b-fair: Learning fair representations with statistical testing, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2023, pp. 9564–9576.

[29] C. Zhao, F. Chen, B. Thuraisingham, Fairness-aware online meta-learning, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 2294–2304.

[30] T. Shi, Y. Zhang, J. Zhang, F. Feng, X. He, Fair recommendations with limited sensitive attributes: A distributionally robust optimization approach, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, pp. 448–457.

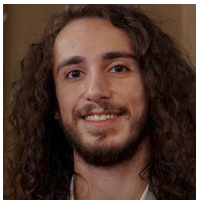
[31] S. Park, S. Hwang, D. Kim, H. Byun, Learning disentangled representation for fair facial attribute classification via fairness-aware information alignment, in: Proceedings of the AAAI conference on artificial intelligence, vol. 35, 2021, pp. 2403–2411.

[32] A. Kraskov, H. Stögbauer, P. Grassberger, Estimating mutual information, Phys. Rev. E–Stat. Nonlinear Soft Matter Phys. 69 (6) (2004) 066138.

[33] M. I. Belghazi, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, A. Courville, R. D. Hjelm, Mine: mutual information neural estimation, arXiv preprint arXiv:1801.04062, 2018.

[34] P. Cheng, W. Hao, S. Dai, J. Liu, Z. Gan, L. Carin, Club: A contrastive log-ratio upper bound of mutual information, in: International conference on machine learning, PMLR, 2020, pp. 1779–1788.
[35] M. Wu, C. Zhuang, M. Mosse, D. Yamins, N. Goodman, On mutual information in contrastive learning for visual representations, arXiv preprint arXiv:2005.13149, 2020.
[36] X. Nguyen, M. J. Wainwright, M. I. Jordan, Estimating divergence functionals and the likelihood ratio by convex risk minimization, IEEE Trans. Inf. Theory 56 (11) (2010) 5847–5861.
[37] Y. Mroueh, I. Melnyk, P. Dognin, J. Ross, T. Sercu, Improved mutual information estimation, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 9009–9017.
[38] F. Agakov, D. Barber, Variational information maximization for neural coding, in: International Conference on Neural Information Processing, Springer, 2004, pp. 543–548.
[39] C. O. Sakar, O. Kursun, A method for combining mutual information and canonical correlation analysis: predictive mutual information and its use in feature selection, Expert Syst. Appl. 39 (3) (2012) 3333–3344.
[40] D. Tsur, Z. Goldfeld, K. Greenewald, Max-sliced mutual information, Adv. Neural Inf. Process. Syst. 36 (2023) 80338–80351.
[41] S. Forrest, Genetic algorithms, ACM Comput. Surv. 28 (1) (1996) 77–80.
[42] E. Barbierato, A. Pozzi, D. Tessera, Controlling bias between categorical attributes in datasets: A two-step optimization algorithm leveraging structural equation modeling, IEEE Access (2023).
[43] P. Van der Laan, The 2001 Census in the Netherlands: Integration of Registers and Surveys, 2001, pp. 39–52.
[44] Y. Gorishniy, I. Rubachev, V. Khrulkov, A. Babenko, Revisiting deep learning models for tabular data, Adv. Neural Inf. Process. Syst. 34 (2021) 18932–18943.
[45] F. Kamiran, A. Karim, X. Zhang, Decision theory for discrimination-aware classification, in: 2012 IEEE 12th international conference on data mining, IEEE, 2012, pp. 924–929.
[46] B. H. Zhang, B. Lemoine, M. Mitchell, Mitigating unwanted biases with adversarial learning, in: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society, 2018, pp. 335–340.
[47] Z. Goldfeld, K. Greenewald, Sliced mutual information: A scalable measure of statistical dependence, Adv. Neural Inf. Process. Syst. 34 (2021) 17567–17578.
[48] A. Incremona, G. De Nicolao, Short-term forecasting of the italian load demand during the easter week, Neural Comput. Appl. (2022) 1–15.
[49] A. Celikkanat, Y. Shen, F. D. Malliaros, Multiple kernel representation learning on networks, IEEE Trans. Knowl. Data Eng. 35 (6) (2022) 6113–6125.
[50] V. Genre, G. Kenny, A. Meyler, A. Timmermann, Combining expert forecasts: Can anything beat the simple average?, Int. J. Forecast. 29 (1) (2013) 108–121.
[51] A. Incremona, G. De Nicolao, F. Fusco, B. J. Eck, S. Tirupathi, Aggregation of non-linearly enhanced experts with application to electricity load forecasting, Appl. Soft Comput. 112 (2021) 107857.
[52] Y. Zeng, X. Yang, L. Chen, C. Ferrer, M. Jin, M. Jordan, R. Jia, Fairness-aware meta-learning via nash bargaining, Adv. Neural Inf. Process. Syst. 37 (2024) 83235–83267.
[53] S. Stan, M. Rostami, Preserving fairness in ai under domain shift, J. Artif. Intell. Res. 81 (2024) 907–934.
[54] H. Wu, X. Luo, M. Zhou, Advancing non-negative latent factorization of tensors with diversified regularization schemes, IEEE Trans. Serv. Comput. 15 (3) (2020) 1334–1344.
[55] H. Wu, Y. Qiao, X. Luo, A fine-grained regularization scheme for nonnegative latent factorization of high-dimensional and incomplete tensors, IEEE Trans. Serv. Comput. (2024).
[56] X. Luo, H. Wu, Z. Li, Neulft: A novel approach to nonlinear canonical polyadic decomposition on high-dimensional incomplete tensors, IEEE Trans. Knowl. Data Eng. 35 (6) (2022) 6148–6166.
[57] R. Nagpal, A. Khan, M. Borkar, A. Gupta, A multi-objective framework for balancing fairness and accuracy in debiasing machine learning models, Mach. Learn. Knowl. Extr. 6 (3) (2024) 2130–2148.
[58] I. Hounie, A. Ribeiro, L. F.O. Chamon, Resilient constrained learning, Adv. Neural Inf. Process. Syst. 36 (2023) 71767–71798.
[59] J. Chai, X. Wang, Fairness with adaptive weights, in: International conference on machine learning, PMLR, 2022, pp. 2853–2866.

Author biography



Alessandro Incremona received a Bachelor's degree and a Master's degree in Computer Engineering from the University of Pavia in 2015 and 2017, respectively. He also earned a Ph.D. in Electronics, Informatics, and Electrical Engineering from the same university in 2021, after which he spent a year there as a Postdoctoral Researcher. In 2019, he held a position as a research intern at IBM Research Dublin. In 2024, he joined the Catholic University of the Sacred Heart in Brescia, Italy, as a Postdoctoral Researcher, and since January 2025, he has been serving as an Assistant Professor in the Faculty of Mathematical, Physical, and Natural Sciences at the same institution. His research interests include Regularized Machine Learning, Imitation Learning, Time Series Forecasting, and Statistical Learning.

Andrea Pozzi earned his Bachelor's degree in Industrial Engineering and Master's degree in Electrical Engineering in 2015 and 2017, respectively, both from the University of Pavia. He also completed his Ph.D. in Electronics, Informatics, and Electrical Engineering at the same institution in 2021. He has held visiting scholar positions at TU Braunschweig in 2016 and UC Berkeley in 2019. After serving as a Postdoctoral Researcher at the University of Pavia, he joined the Catholic University of Sacred Heart in Brescia, Italy, as an Assistant Professor of Machine Learning in the Faculty of Mathematical, Physical, and Natural Sciences in January 2022. His research interests encompass Reinforcement Learning, Imitation Learning, Machine Learning, Approximate Dynamic Programming, and Advanced Control Theory.

Alessandro Guiscardi earned a Bachelor's degree in Economics in 2021 from the University of Brescia and a Master's degree in Applied Data Science for Banking and Finance in 2022 from the Catholic University of the Sacred Heart in Brescia. In 2022, he was awarded the Gemelli Prize in recognition of his academic merit.

Daniele Tessera received his Ph.D. in Computer Engineering from the University of Pavia, Italy. He began his academic career at the Catholic University of the Sacred Heart in Brescia, Italy, where he served as Assistant Professor from 2003 and was promoted to Associate Professor in 2011 at the Faculty of Mathematical, Physical, and Natural Sciences. In 2025, he joined the Department of Electrical, Computer and Biomedical Engineering at the University of Pavia, Italy. His research interests focus on performance debugging and benchmarking, workload characterization, cloud computing, and the application of machine learning techniques. He is the co-author of more than 40 papers published in international journals and conference proceedings.