Maximal Correlation Feature Selection and Suppression With Applications

by

Joshua Ka-Wing Lee

B.A.Sc., University of Toronto (2015)
S.M., Massachusetts Institute of Technology (2017)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2021

Author . . . .
Department of Electrical Engineering and Computer Science
October 22, 2021

Certified by. . . . .
Gregory W. Wornell
Sumitomo Professor Engineering
Thesis Supervisor

Accepted by . . . .
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Committee on Graduate Students

Maximal Correlation Feature Selection and Suppression With Applications

by

Joshua Ka-Wing Lee

Submitted to the Department of Electrical Engineering and Computer Science
on October 22, 2021, in partial fulfillment of the
requirements for the degree of
Doctor of Science in Electrical Engineering and Computer Science

Abstract

In standard supervised learning, we assume that we are trying to learn some target variable Y from some data X. However, many learning problems can be framed as supervised learning with an auxiliary objective, often associated with an auxiliary variable D which defines this objective. Applying the principles of Hirschfeld-Gebelein-Rényi (HGR) maximal correlation analysis reveals new insights as to how to formulate these learning problems with auxiliary objectives. We examine the use of the HGR in feature selection for multi-source transfer learning learning in the few-shot setting. We then apply HGR to the problem of feature suppression via enforcing marginal and conditional independence criteria with respect to a sensitive attribute, and illustrate the effectiveness of our methods to problems of fairness, privacy, and transfer learning. Finally, we explore the use of HGR in extracting features for outlier detection.

Thesis Supervisor: Gregory W. Wornell
Title: Sumitomo Professor Engineering

Acknowledgments

My academic journey would not have been possible without the support of the many people who helped me grow as both a researcher and as a person, and I would like to extend my most sincerest thanks to them.

First and foremost, I would like to thank my advisor, Professor Gregory W. Wornell, for all of his guidance and teachings in my six wonderful years here at MIT. Beyond the wonderful insights and education I've received from him, I am also incredibly grateful for his understanding and for providing me the support I needed at each stage of my journey. Much of my growth as a student and a researcher can be traced back to Professor Wornell, and I will never forget that.

Professor Lizhong Zheng has also been there since very early in my graduate career, and has always been ready to provide an alternative perspective on any problem I might have, as well as to encourage me to dig deep and better understand the tools I was using. Without his guidance, I would not have developed an appreciation for staying grounded even when exploring the farthest reaches of knowledge space.

Dr. Prasanna Sattigeri was there for me when I stepped outside of the ivory towers of academia and began to further my research out in the wider world. In addition to his invaluable theoretical insights, he also taught me how to think in terms of practical applications, providing me with the opportunities to see my work take shape in industry, and for that, I will always be grateful.

In addition to my committee, I am also grateful towards the many others who have been with me in the past six years. I am honoured to have been able to work with Professor Devavrat Shah and David Qiu, and I value the fruits of my discussions with them, as well as with Anuran Makur and Matthew Staib.

Much of my work would also have not been possible without all the wonderful people at the MIT-IBM Watson AI Lab supporting me in various ways, providing both mentorship, resources, and ideas that helped make this thesis the document it is today. To Prasanna Sattigeri, Rameswar Panda, Deepta Rajan, Subhro Das, Kate Soule, and Mark Weber, thank you so very much for giving me the opportunity to

work with such a wonderful team, and I hope the initiatives I was fortunate enough to take part in continue to flourish.

The community at SIA has also been vital for my success. I count myself so very lucky to have been able to have found a home away from home in this group, and I will dearly miss all the good times and brilliant discussions we had within the walls of that office on the sixth floor.

To Yuheng Bu, thank you for being a wonderful collaborator, and for helping me learn how to improve my academic writing. To Tricia O'Donnell, thank you for keeping the lab running smoothly and for going above and beyond to create a welcoming environment for us all. And to Ying-Zong Huang, Wai Lok Lai, Atulya Yellepeddi, Qing He, Ganesh Ajjanagadde, Gal Shulkind, Gauri Joshi, Xuhong Zhang, Christos Thrampoulidis, Ankit Singh Rawat, Amichai Painsky, Toros Arikan, Tejas Jayashankar, Gary Lee, Safa Medin, Abhin Shah, Adam Yedidia, Mumin Jin, Isabella Kang, Samuel Tenka, Tony Tong Wang, and Xiaoyi Wang, thank you so much for being wonderful labmates, and my only regret is that I could not spend as much time with you all as I would have liked.

Beyond academia, I would also like to thank the members of the Diamond Sparkle Foundation for their moral support as well as their assistance in helping me improve in my writing ability, which has been vital to my success.

Finally, I extend my deepest thanks and love to my parents, who have made all of this possible, and shaped me into the kind of person who could undertake this journey. I will treasure their love, support, and teachings for the rest of my days.

Nai nár andúneo tulyauva vë tintilaila undómenna, ar nai tuvualwë i tinwírion anafaila mettassë lendalvo.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

We are living in an age of unprecedented growth in artificial intelligence research.

The rapid increase in the number of peer-reviewed AI papers has been noted by many in the field [77, 11, 117], and is illustrated in Figure 1-1. However, more than the raw number of papers, new applications for AI are being encountered at an unprecedented rate.

The breadth of problems that machine learning is expected to be able to solve is growing with each passing day, and it is becoming increasingly important to tackle them in a timely fashion. Thus, more than ever, there is a great need to unify as many of the myriad forms that machine learning problems can take under as few paradigms as possible in order to facilitate faster formulation of solutions and allow insights from different fields to diffuse through the learning community.

1.1 Endless Forms Most Beautiful

The standard bivariate supervised learning framework has existed in some form or another for as long as the ideas of decision-making and estimation have been around [17]. In its most general form, it asks us to predict some value Y from some data X, with some cost C associated with the prediction that is generally higher the more wrong we are about the prediction. To aid in this process, we are also given some information about the relationship between X and Y, usually in the form of samples

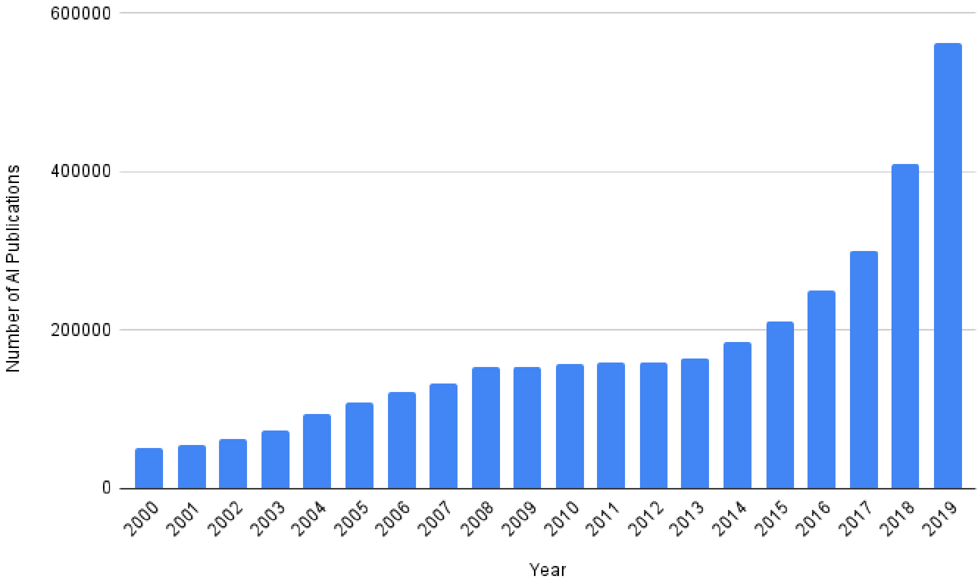


Figure 1-1: Number of peer-reviewed AI publications released each year. Source: [134].

drawn from the joint distribution $P_{X,Y}$ between the two variables.

This framework covers everything from linear regression (which dates back to the early 1800s [72]) to modern image classification with deep neural networks [71]. The applications of these techniques are too numerous to list, and yet, they do not span the range of all possible learning problems we face today.

In particular, as our applications grow more complex, we often run into the problem of multiple competing objectives. This can come in the form of additional constraints on learning Y, such as requiring that a predictive system also be able to identify outliers in a system.

Often times, we also have additional variables that affect our learning objective, possibly introducing new constraints. For example, in fair machine learning, we have an auxiliary variable such as gender or race that we wish our predictions to be fair with respect to. In cases such as these, we also encounter the phenomenon whereby our objectives may conflict with one another, e.g., a classifier that is more fair may perform worse on the main task of predicting Y.

Further complicating this situation is the ever-shifting landscape of machine learning. Not only are new problems uncovered over the course of technological evolution, but societal perspectives and needs with regards to existing problems shift as well. For example, our notion of what constitutes a fair and unbiased decision system may change, from requiring one set of fairness conditions on one group to requiring a different set of conditions on another group. The advent of very large datasets and distributed learning has also placed a greater importance on faster algorithms that require less interconnection between components to function.

In some cases, finding a solution to these problems is also very time-critical. While a gap between conception of a problem and formulation of a solution might only result in a delay for a product going to market, if an existing system is found to be biased towards certain protected groups, that system might still continue to be used in the absence of any solutions to correct the bias. For example, the COMPAS recidivism score, designed to predict whether or not a convicted criminal would reoffend [91], was found to be biased against black individuals [3]. Yet, the system continued to be used in determining criminal sentencing [128]. Thus, being able to quickly find effective and easily-implementable solutions is also of utmost importance to ensure that such injustices are stopped sooner rather than later.

Given the wide variety of secondary learning objectives, designing bespoke, ad-hoc solutions from scratch for every single one of them is impractical at best, and impossible at worst given the time frames allowed for development. It would thus be beneficial to have some central perspective for looking at them in a way that translates into easily being able to derive algorithms for learning under these objectives.

In addition, for this perspective to be truly universal, it must be applicable to data and problems of any modality. Sometimes, our data X might be discrete or categorical (e.g., census and other demographic data, text) and other times, they may be continuous (e.g., images, audio). In addition, our target Y might be discrete (classification) or continuous (regression). We would like to be able to handle all of these cases as they arise.

1.2 Feature Extraction, Selection, and Suppression

A good starting point for studying this problem of modelling is to look deeper into the concepts of feature extraction and selection. Learning useful representations of data is a central principle in machine learning, and thus, by integrating our secondary objectives into the process of learning features of the data, we can control the balance between these objectives.

Loosely speaking, we can view feature extraction as the process by which we learn some function $\Theta(X)$ from the data, the outputs of which are our *features* [14], and are denoted as $U = \Theta(X)$. Feature selection, on the other hand, assumes that we already have access to $\Theta(X)$, but only wish to use a subset of them for prediction (alternatively, we may be interested in "soft" feature selection whereby we reweight the features in order to modulate their importance in making the final prediction) [67]. This can arise because we are using them for a separate task and wish to find the features that are most relevant for the new task [98, 115]. Alternatively, we may wish to select features that obey some secondary constraint, such as fairness [47], or to select the most effective features for the same task but with a bottleneck constraint on the number of features that we can keep (for example, if they are being sent to another device for further processing) [101].

In the process of both feature extraction and selection, we can also attempt to *suppress* certain features. For example, we may desire that information extracted from a user's travel history in a contact-tracing app be decoupled from their identity in order to preserve privacy, and thus any uniquely identifying features should not be represented in the features [12].

Under all of these views, we can see that secondary objectives alter the normal flow of these processes by introducing some new constraint or penalty defined by some dependence between variables.

1.3 Measures of Dependencies

Thus, we need a way to quantify the dependency between random variables. For the sake of practicality, this measure must be easily computable and allow for the derivation of feature learning algorithms that can be implemented, tested, and applied to real datasets. However, it must also be rich and able to capture a wide range of dependencies.

Basic linear measures of dependence, such as correlation $\mathbb{E}[XY]$ or covariance $\mathbb{E}[(X-\mu_X)(Y-\mu_Y)]$ satisfy the practicality conditions, as computations of their estimates require only a single pass through the data. Their simplicity also makes them easy to interpret. However, while they may work well for Gaussian data, they are very weak in their ability to capture the full range of nonlinear dependencies between variables, and are generally not relevant when the random variables are categorical.

On the other end, mutual information is a popular measure of dependence between two variables that is incredibly rich in its ability to capture the relationship between variables. The mutual information between random variables X and Y is defined as $I(X;Y) = \mathbb{E}_{P_{X,Y}}\left[\log \frac{P_{X,Y}(X,Y)}{P_X(X)P_Y(Y)}\right]$ [30].

However, mutual information has practical drawbacks that can make it difficult to use. It is simple enough to estimate and optimize for in the discrete case, but in the continuous case, estimating the mutual information for continuous data in a way that allows for optimization requires either binning the data into discrete bins, which introduces quantization issues and reduces universality, in addition to not scaling well when the number of dimensions is large, or requires some kind of parametric estimate for $P_{X,Y}$ (e.g., variational methods). In the latter case, many of these methods for estimating the probability (such as variational autoencoders) suffer from problems of instability [110, 75].

Mutual information also suffers from an interpretability issue, where even if one has a good estimate of the mutual information, this is not necessarily indicative of how X and Y depend on one another.

Ideally, we would like a method that is simple, yet powerful, able to capture all possible dependencies between variables, but still computable and interpretable.

A solution to this dilemma comes to us through the Hirschfeld-Gebelein-Rényi (HGR) maximal correlation, a nonlinear measure of dependence that also yields a set of feature functions, which provide interpretability, and can be computed using a simple algorithm. Furthermore, in certain cases where the dependence between the random variables are weak, one can also draw connections to other measures of dependence.

This measure also provides a powerful paradigm for viewing learning problems and devising solutions to them, allowing us to adapt to new settings more quickly. In the following sections, we will illustrate the breadth of problems that can be solved with the HGR, and the effectiveness of the solutions derived.

Chapter 2

The Hirschfeld-Gebelein-Rényi Maximal Correlation

As stated in the previous chapter, one of the measures of dependency that contains the properties we desire is the HGR maximal correlation. In this chapter, we will present a brief overview of this measure, including the properties we will make use of and the methods for computing this quantity and those related to it.

We also note here that this chapter mostly summarizes results from [59], which contains a much more thorough treatment of the properties of the HGR.

2.1 Definition

We begin by defining the HGR maximal correlation [56, 40, 103]:

Definition 1 (Hirschfeld-Gebelein-Rényi Maximal Correlation)**.** *Let $X \in \mathcal{X}, Y \in \mathcal{Y}$ be jointly distributed random variables. Then*

$$\mathrm{HGR}(X;Y) = \sup_{\substack{f:\mathcal{X}\to\mathbb{R},g:\mathcal{Y}\to\mathbb{R} \\ \mathbb{E}[f(X)]=\mathbb{E}[g(Y)]=0 \\ \mathbb{E}[f^2(X)]=\mathbb{E}[g^2(Y)]=1}} \mathbb{E}[f(X)g(Y)] \tag{2.1}$$

is the ***HGR maximal correlation*** *between X and Y, and f, g the associated* ***maximal correlation functions****.*

Some of the useful basic properties of the HGR maximal correlation are:

- $0 \leq \mathrm{HGR}(X;Y) \leq 1$.
- $\mathrm{HGR}(X;Y) = 0$ if and only if X and Y are independent.
- $\mathrm{HGR}(X;Y) = 1$ if and only if there exists f and g such that $f(X) = g(Y)$ with probability 1.

Thus, it is immediately obvious that the HGR can be used as a way of measuring the nonlinear dependence between two random variables, and is very conveniently bounded between 0 and 1, with higher values indicating a greater degree of dependence.

2.2 The Divergence Transfer Matrix

When X and Y are both discrete, we can take a linear algebraic view of the HGR by analyzing an equivalent representation of the joint distribution $P_{X,Y}$ known as the divergence transfer matrix (DTM) [61].

Definition 2 (Divergence Transfer Matrix)**.** *The (y,x)th entry of the DTM $\mathbf{B}_{X,Y} \in \mathbb{R}^{|\mathcal{Y}|\times|\mathcal{X}|}$ associated with joint distribution $P_{X,Y}$ is given by $\mathbf{B}_{X,Y}(x,y) \triangleq \frac{P_{Y,X}(y,x)}{\sqrt{P_Y(y)}\sqrt{P_X(x)}}$.*

The DTM can always be computed from $P_{X,Y}$, and possesses some useful properties related to the HGR [69].

In particular, the first singular value σ_0 of the DTM $\mathbf{B}_{X,Y}$ is 1, and its corresponding right and left singular vectors are given by $\phi_0(x) = \sqrt{P_X(x)}$ and $\psi_0(y) = \sqrt{P_Y(y)}$, respectively. Furthermore, the second-largest singular value is equal to the HGR, and the associated associated right and left singular vectors $\phi_1(x), \psi_1(y)$ are related to the maximal correlation functions $f(x), g(y)$ by

$$f(x) = \frac{\phi_1(x)}{\sqrt{P_X(x)}}, \quad g(y) = \frac{\psi_1(y)}{\sqrt{P_X(y)}}. \tag{2.2}$$

Thus, computing the HGR and the corresponding maximal correlation functions reduces to performing a singular value decomposition (SVD) of the DTM [58].

2.3 The k-Mode HGR Maximal Correlation

Using a DTM-based approach to the HGR, we can now extend its definition to encapsulate the additional singular values beyond the top two.

Definition 3 (k-mode HGR)**.** *Given $1 \leq k \leq K-1$ with $K = \min\{|\mathcal{X}|, |\mathcal{Y}|\}$, the k-mode maximal correlation problem for random variables $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ is*

$$(\mathbf{f}^*, \mathbf{g}^*) \triangleq \underset{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k,\ \mathbf{g}\colon \mathcal{Y}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbb{E}[\mathbf{g}(Y)]=\mathbf{0}, \\ \mathbb{E}\left[\mathbf{f}(X)\mathbf{f}^{\mathrm{T}}(X)\right]=\mathbb{E}\left[\mathbf{g}(Y)\mathbf{g}^{\mathrm{T}}(Y)\right]=\mathbf{I}}}{\arg\max} \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\,\mathbf{g}(Y)\right], \tag{2.3}$$

where expectations are with respect to joint distribution $P_{X,Y}$. We refer to $\mathbf{f}^*$ and $\mathbf{g}^*$ as the maximal correlation functions. With $\mathbf{f}^* = (f_1^*, \ldots, f_k^*)^{\mathrm{T}}$ and $\mathbf{g} = (g_1^*, \ldots, g_k^*)^{\mathrm{T}}$, we further define the associated maximal correlations $\sigma_i = \mathbb{E}\left[f_i^*(X)\, g_i^*(Y)\right]$ for $i = 1, \ldots, k$.

This approach to feature extraction provides us with a hierarchy of orthogonal features sorted by the amount of dependence they capture between X and Y. The solution to this problem is also given to us via SVD of the DTM [58].

Theorem 1. *[103, 69] The first singular value of the DTM $\mathbf{B}_{X,Y}$ is 1, and its corresponding right and left singular vectors are given by $\phi_0(x) = \sqrt{P_X(x)}$ and $\psi_0(y) = \sqrt{P_Y(y)}$, respectively. Furthermore, the next k largest singular values are the k maximal correlations associated with the k-mode HGR problem, and the associated associated right and left singular vectors $\phi_i(x), \psi_i(y)$ for the ith singular value are related to the ith maximal correlation functions $f_i^*(x), g_i^*(y)$ by*

$$f_i^*(x) = \frac{\phi_i(x)}{\sqrt{P_X(x)}}, \quad g_i^*(y) = \frac{\psi_i(y)}{\sqrt{P_X(y)}}. \tag{2.4}$$

In addition, this SVD characterization induces the following modal decomposition of the joint distribution:

$$P_{X,Y}(x,y) = P_X(x)\, P_Y(y) \left[1 + \sum_{i=1}^{K-1} \sigma_i\, f_i^*(x)\, g_i^*(y) \right], \tag{2.5}$$

via which predictions are made according to

$$P_{Y|X}(y|x) = P_Y(y) \left(1 + \sum_{i=1}^{K-1} \sigma_i f_i^*(x) g_i^*(y) \right). \tag{2.6}$$

Thus, $\mathbf{f}^*$ can be viewed as a sufficient statistic for Y from X (and vice versa for $\mathbf{g}^*$).

2.4 Local Approximations

For another useful interpretation of the HGR, we first introduce the concept of ϵ-*neighbourhoods* and ϵ-*dependence* [58].

Definition 4 (ϵ-neighbourhood)**.** *For a given $\epsilon > 0$, the ϵ-neighbourhood of a distribution $P \in \mathsf{relint}(\mathcal{P}^{\mathcal{X}})$ is defined as*

$$\mathcal{N}_\epsilon^{\mathcal{Z}}(P) \triangleq \{P' \in \mathcal{P}^{\mathcal{X}} | D_{\chi^2}(P'||P) \leq \epsilon^2\} \tag{2.7}$$

where, for $P \in \mathcal{P}^{\mathcal{X}}$ and $Q \in \mathsf{relint}(\mathcal{P}^{\mathcal{X}})$,

$$D_{\chi^2}(P||Q) \triangleq \sum_{x \in \mathcal{X}} \frac{(Q(x) - P(x))^2}{Q(x)}, \tag{2.8}$$

and $\mathcal{P}^{\mathcal{X}}$ is the space of probability distribution over finite alphabet $\mathcal{X}$ and $\mathsf{relint}(\mathcal{P})$ is the relative interior of $\mathcal{P}$.

Definition 5 (ϵ-dependence)**.** *Let $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ be random variables with joint distribution $P_{X,Y} \in \mathcal{P}^{\mathcal{X}\times\mathcal{Y}}$, where we restrict $\mathcal{P}^{\mathcal{X}\times\mathcal{Y}}$ to contain all valid joint distributions with strictly positive marginals (which we denote with P_X and P_Y, respectively). Then, X and Y are ϵ-dependent if there exists $\epsilon > 0$ such that $P_{X,Y} \in \mathcal{N}_\epsilon^{\mathcal{X}\times\mathcal{Y}}(P_X P_Y)$.*

Then, we have the following result that is particularly useful when X and Y are weakly dependent (i.e., ϵ is small) [59].

Lemma 2. *Let X and Y be ϵ-dependent random variables. Then:*

$$I(X;Y) = \frac{1}{2}||\mathbf{B}_{X,Y}||_F^2 - \frac{1}{2} + o(\epsilon^2) = \frac{1}{2}\sum_{i=1}^{K-1}\sigma_i^2 - \frac{1}{2} + o(\epsilon^2) \tag{2.9}$$

where $||\mathbf{A}||_F$ denotes the Frobenius norm of matrix $\mathbf{A}$.

Thus, we can draw connections between the HGR and mutual information, which can allow for certain approximations to be derived, as well as provide another justification for why one might wish to use the HGR to measure and modulate the relationship between random variables.

An additional useful extension of Lemma 2 is that, for $1 < k < K-1$, for sufficiently small ϵ, we can construct a k-mode estimate for the joint distribution between X and Y as follows:

$$P_{X,Y}^{(k)}(x,y) \triangleq P_X(x)\,P_Y(y)\left[1 + \sum_{i=1}^{k}\sigma_i\, f_i^*(x)\, g_i^*(y)\right], \tag{2.10}$$

If we denote $X^{(k)}$ and $Y^{(k)}$ as the variables induced by $P_{X,Y}^{(k)}$, then we also have that:

$$I(X^{(k)};Y^{(k)}) = \frac{1}{2}\sum_{i=1}^{k}\sigma_i^2 - \frac{1}{2} + o(\epsilon^2) \tag{2.11}$$

The immediate consequence of this is that the kth feature pair contributes $\frac{1}{2}\sigma_k^2 + o(\epsilon^2)$ to the overall mutual information, which gives us a useful way to sort these features. By arranging them in order of the magnitude of σ_k, we are implicitly sorting by how much information they capture between X and Y.

2.5 Variational Characterizations

The maximal correlation functions can also be obtained via a variational characterization of the SVD, which in turns provides an alternative optimization for the HGR that will be used later.

In particular, for a DTM $\mathbf{B}_{X,Y}$, the following optimization yields the singular vectors of $\mathbf{B}_{X,Y}$ [58], along with the approximation for the mutual information $I(X;Y)$:

$$I(X;Y) \approx \sum_{i=0}^{k} \sigma_i^2 = \max_{\mathbf{\Phi}_X^{\mathrm{T}} \mathbf{\Phi}_X = \mathbf{I}} \|\mathbf{B}_{X,Y} \mathbf{\Phi}_X\|_{\mathrm{F}}^2. \tag{2.12}$$

Using the property that $\|\mathbf{A}\|_{\mathrm{F}}^2 = \mathrm{tr}(\mathbf{A}\mathbf{A}^T)$, where $\mathrm{tr}(\mathbf{A})$ denotes the trace of matrix $\mathbf{A}$, this optimization problem can be rewritten as

$$\sum_{i=0}^{k} \sigma_i^2 = \max_{\mathbf{\Phi}_X^{\mathrm{T}} \mathbf{\Phi}_X = \mathbf{I}} \mathrm{tr}(\mathbf{\Phi}_X^{\mathrm{T}} \mathbf{B}_{X,Y}^{\mathrm{T}} \mathbf{B}_{X,Y} \mathbf{\Phi}_X), \tag{2.13}$$

which can be solved via an eigendecomposition of $B_{X,Y}^{\mathrm{T}} B_{X,Y}$.

If we rewrite this optimization instead using $P_{X,Y}$ and $\mathbf{f}(x)$, we obtain the following expression instead:

$$\sum_{i=0}^{k} \sigma_i^2 = \max_{\substack{\mathbf{f}\colon \mathcal{X} \to \mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)] = \mathbf{0} \\ \mathbb{E}\left[\mathbf{f}(X)\mathbf{f}^{\mathrm{T}}(X)\right] = \mathbf{I}}} \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2 \right] \tag{2.14}$$

where $\|\mathbf{f}\|$ denotes the L^2 norm of $\mathbf{f}$.

2.6 Computing the Maximal Correlation

As we have seen, the HGR is a useful tool for measuring the dependence between two variables, providing both interpretable features and a connection to another popular measure of dependence (mutual information). However, its computation requires finding the optimal maximal correlation functions.

In the case where X and Y are discrete, this can be done by finding the singular values and singular vectors of the DTM. In the continuous case, we can use neural networks to approximate the maximal correlation functions instead.

We also note here that in general, we will not have access to $P_{X,Y}$ directly, but rather an empirical distribution $\hat{P}_{X,Y}$ estimated from samples drawn from $P_{X,Y}$. However, estimating the modes of the HGR is sample efficient (error exponent grows linearly with number of modes) as long as the number of modes is small [58, 60, 84].

2.6.1 The Alternating Conditional Expectation Algorithm

The *power method* and its most direct generalization, *orthogonal iteration*, is one of the oldest methods for computing the singular value decomposition of a matrix [116]. When applied to the DTM, we derive the following expressions relating the maximal correlation functions f and g [58]:

$$\mathbb{E}_{p_{X|Y}(\cdot|y)}\left[f_i^*(X)\right] = \sigma_i\, g_i^*(y) \quad \text{and} \quad \mathbb{E}_{p_{Y|X}(\cdot|x)}\left[g_i^*(Y)\right] = \sigma_i\, f_i^*(x),$$

This leads us to the alternating conditional expectations (ACE) algorithm of Breiman and Friedman [16] for computing these functions. Indeed, for a given $\mathbf{f}$, the correlation maximizing $\mathbf{g}$ has components

$$g_i^*(y) \propto \mathbb{E}_{p_{X|Y}(\cdot|y)}\left[f_i^*(X)\right], \quad i = 1, \ldots, k. \tag{2.15}$$

and vice-versa. The ACE algorithm is described in Algorithm 1.

2.6.2 The Soft-HGR

However, although the ACE algorithm can be applied to continuous X and Y, the immediate problem that arises is defining the space of functions to search for the maximal correlation functions f^* and g^*.

We can use deep neural networks to define this search space and therefore approximate the maximal correlation functions, exploiting their universality [57]. This is especially practical as most machine learning systems that operate on highly nonlinear data use such networks for inference already.

Secondly, we come to the problem of learning these functions. The ACE algorithm performs poorly in the continuous case as computing the conditional expectation updates on samples requires some type of smoothing algorithm, which may be challenging to implement with particularly high-dimensional data (e.g., the space of

Algorithm 1 The Alternating Conditional Expectation (ACE) Algorithm for multiple mode extraction

```
Data: Joint distribution $P_{X,Y}$ (or empirical $\hat{P}_{X,Y}$ estimated from samples) and number of modes to extract $k$.
Result: Associated maximal correlations $\sigma^* = \{\sigma_1, ..., \sigma_k\}$ and correlation functions $\mathbf{f}^*(x) = \{f_1(x), ..., f_k(x)\}, \mathbf{g}^*(y) = \{g_1(y), ..., g_k(y)\}$
Initialize $\mathbf{g}^*(y)$ with random values.
repeat
    $\forall x : \mathbf{f}^*(x) \leftarrow \mathbb{E}\left[\mathbf{g}^*(Y)|X = x\right]$
    $\forall x : \mathbf{f}^*(x) \leftarrow \mathbf{f}^*(x) - \mathbb{E}\left[\mathbf{f}^*(x)\right]$ // center
    Factor using QR Decomposition: $\mathbf{f}^*\sqrt{\mathbf{P_X}} = \mathbf{QR}$
    Orthogonalize: $\mathbf{f}^* \leftarrow \mathbf{f}^*\mathbf{R}$
    $\forall y : \mathbf{g}^*(y) \leftarrow \mathbb{E}\left[\mathbf{f}^*(X)|Y = y\right]$
    $\forall y : \mathbf{g}^*(y) \leftarrow \mathbf{g}^*(y) - \mathbb{E}\left[\mathbf{g}^*(y)\right]$ // center
    Factor using QR Decomposition: $\mathbf{g}^*\sqrt{\mathbf{P_Y}} = \mathbf{QR}$
    Orthogonalize: $\mathbf{g}^* \leftarrow \mathbf{g}^*\mathbf{R}$
    $\forall n = 1, ..., k : \sigma_n \leftarrow \mathbb{E}\left[f_n(X)g_n(Y)\right]$
until $\sigma^*$ stops increasing;
return $\mathbf{f}^*(x), \mathbf{g}^*(y), \sigma^*$
```

images), and constrains the search space according to the smoothing used.

In this case, we can directly learn the functions by optimizing the maximal correlation objective:

$$(\mathbf{f}^*, \mathbf{g}^*) = \underset{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k,\ \mathbf{g}\colon \mathcal{Y}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbb{E}[\mathbf{g}(Y)]=\mathbf{0}, \\ \mathbb{E}\left[\mathbf{f}(X)\mathbf{f}^{\mathrm{T}}(X)\right]=\mathbb{E}\left[\mathbf{g}(Y)\mathbf{g}^{\mathrm{T}}(Y)\right]=\mathbf{I}}}{\arg\max} \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\,\mathbf{g}(Y)\right]. \tag{2.16}$$

However, the orthogonality constraint is difficult to implement, usually requiring a whitening process that, in the continuous case for high-dimensional data, has high complexity and stability issues. Our solution here is to instead use a variational approximation of the HGR, known as the soft-HGR [119]. The soft-HGR relaxes the orthogonality constraint using an alternative approach to solving the SVD by finding a low-rank approximation of the DTM $\mathbf{B}_{X,Y}$:

$$\mathrm{HGR}_{\mathrm{soft}}(X;Y) \triangleq \max_{\substack{\mathbb{E}[\mathbf{f}(X)]=\mathbf{0}\\ \mathbb{E}[\mathbf{g}(Y)]=\mathbf{0}}} \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\,\mathbf{g}(Y)\right] - \frac{1}{2}\,\mathrm{tr}(\mathrm{cov}(\mathbf{f}(X))\,\mathrm{cov}(\mathbf{g}(Y))). \tag{2.17}$$

This formulation also has the advantage of being easily differentiable for use in systems that learn via gradient descent (e.g., neural networks).

2.7 Concluding Remarks

In this chapter, we have introduced a measure of dependence between variables that is interpretable, adaptable to both discrete and continuous settings, and computable using simple objectives or algorithms. In addition, it can approximate the mutual information in the weakly dependent case, and also allows for the derivation of a modal decomposition of the joint probability distribution that can be used for prediction in all cases.

In the upcoming chapters, we will explore the various ways in which the HGR formulations can be used to model different learning objectives to solve a wide variety of problems, and examine the performance of these solutions on a wide range of applications.

Chapter 3

Multi-Source Transfer Learning

Equipped with the HGR maximal correlation, we now examine the myriad problems it can be used to solve. To begin, we consider a problem in transfer learning that illustrates how an HGR-based approach can be used to develop an algorithm for feature selection. In particular, we examine the multi-source transfer learning problem, and develop a method to use the HGR to select and evaluate existing features for transfer.

Recently, the development of efficient algorithms for training deep neural networks on diverse platforms with limited interaction has created both opportunities and challenges for deep learning. An emerging example involves training networks on mobile devices [51, 94, 70]. In such cases, while each user's device may be training on a different set of data with a different classification objective, multi-task learning techniques can be used to leverage these separate datasets in order to transfer to new tasks for which we observe few samples.

However, most existing methods require some aspect of control over the training on the source datasets. Either all the datasets must be located on the same device for training based on some joint optimization criterion, or the overall architecture requires some level of control over the training for each individual source dataset. In the case of, e.g., object classification in images collected by users, sending this data to a central location for processing may be impractical, or even a violation of privacy rights. Alternatively, it is possible that one might wish to use older, pre-trained classifiers for which the original training data is no longer available, and to transfer them for use in a new task. In either case, it could be acceptable to transmit the neural network features learned by the device in an anonymized fashion, and to then combine the features learned by multiple users in order to classify novel images.

This would be an example of a multi-task learning problem in which we have not only multiple source datasets, but access to only pre-trained networks (whose learning objective we cannot control) from those datasets, not the underlying training data used, and we wish to train a classifier for some new target label set given only a few target samples.

Fine-tuning methods can be used when the source network is frozen to transfer to a target domain, but these methods tend not to work very well in a few-shot setting when there are multiple networks due to the number of parameters necessary for fine-tuning, especially in an environment where features cannot be learned with the intention of transfer [36]. In addition, we may also wish to select only a subset of the features to fine-tune on, if we are limited in the number of features that can be queried.

3.1 Problem Setup

Consider a multi-task learning setup in which we have N different source classification tasks $\{\mathcal{T}_1^s, \dots, \mathcal{T}_N^s\}$, for which we have labeled data $\{(x_1^{s_n}, y_1^{s_n}), \dots, (x_{k_n}^{s_n}, y_{k_n}^{s_n})\}$ for task $\mathcal{T}_n^s$, $n \in \{1, \dots, N\}$. We also have a single target task $\mathcal{T}^t$, with associated labeled data $\{(x_1^t, y_1^t), \dots, (x_k^t, y_k^t)\}$.

For this problem we assume that $x_i^{s_n} \in \mathcal{X}$ for all n and i, and $x_i^t \in \mathcal{X}$ for all i, that is, the data for the target and each source task are drawn from some common alphabet (e.g., all data are natural images). We do not assume any overlap between labels for any pair of datasets (i.e., $y_i^{s_n} \in \mathcal{Y}^{s_n}$ for all n and i, and $y_i^t \in \mathcal{Y}^t$ for all i, where $\mathcal{Y}^t \neq \mathcal{Y}^{s_1} \neq \ldots \neq \mathcal{Y}^{s_N}$).

For each source task $\mathcal{T}_n^s$, we have access to a pre-trained neural network that we assume to have been trained to classify y^{s_n} from x^{s_n}. We assume that the network has some number of layers corresponding to the extraction of features from x^{s_n}, followed by a final classification layer that maps the features to a predicted class label $\hat{y}^{s_n}$. We denote the output of the penultimate layer as $\mathbf{f}^{s_n}\colon \mathcal{X} \to \mathbb{R}^{l_n}$, of which the ith feature is $f_i^{s_n} : \mathcal{X} \to \mathbb{R}$, where l_n is the number of features output by this layer. We denote the final layer as $h^{s_n}\colon \mathbb{R}^{l_n} \to \mathcal{Y}^{s_n}$, so that the entire neural network classifier can be written as $\hat{y} = h^{s_n}(\mathbf{f}^{s_n}(x))$.

We seek to train a classifier on the target task given training samples $\{(x_1^t, y_1^t), \ldots, (x_k^t, y_k^t)\}$, with access to h^{s_n} and $\mathbf{f}^{s_n}$ for each source dataset, but without any access to the underlying source training samples $\{(x_1^{s_n}, y_1^{s_n}), \ldots, (x_{k_n}^{s_n}, y_{k_n}^{s_n})\}$.

As an example context, this reflects a situation in which there are many devices collecting and analyzing data, but where the target learner is not allowed to access the data, either because the devices have limited bandwidth and cannot transmit everything they have collected, the data is personal (i.e., pictures taken by users of a mobile app) and cannot be transmitted for privacy purposes, or the original data is otherwise lost (if the data was collected a long time ago). However, in these cases, it may still be possible to query the classifier trained on each device to get their intermediate features, which would require less information to be transmitted.

3.1.1 Prior Work

Multi-task learning is a well-studied problem, with several variations and formulations. One standard approach is to learn a common feature function $f(\cdot)$ across all tasks that optimize some joint objective, followed by a final classification layer for each task [81, 95]. This is a technique that has some theoretical guarantees as given by Ben-David et al. [10]. While effective, this method requires joint training, which our problem formulation precludes.

Gupta and Ratinov [50] propose a method of combining the outputs of multiple pre-trained classifiers by training on their raw predictions, but this method is designed for pre-trained classifiers specially selected to work well in combination with the target task, with an emphasis on cases where the number of possible class labels (i.e., the value of each $|\mathcal{Y}^{s_n}|$) is large, which we do not assume in our problem formulation.

Other methods involve some kind of sequential learning [105] or shared memory unit [78], which could decentralize data storage, but which still require joint control over the training [76].

Meta-learning algorithms have also gained popularity in recent years [87, 90]. These algorithms attempt to learn a suitably general learning rule or model from a set of source tasks that can be fine-tuned with data from a target task [36]. While these methods allow for the combining of multiple source datasets, they are still bound by the need for centralized training.

Finally, the notion of transferring from a single pre-trained network onto a new target task has also been studied before. Yosinski et al. explore the transferability of different layers of a neural net to other tasks in the context of learning general features [129], while Bao et al. propose a score for measuring transferability of features across tasks [7].

3.2 Multi-Source Transfer Learning via Maximal Correlations

In order to perform well on the target task, we seek to model the target $P_T(y|x)$ using an exponential family using the f_i's, for maximal generalizability:

$$P_T(y|x) = P_Y(y) \exp\left(\sum_i \sigma_i f_i(x)\, g_i(y)\right).$$

We have from Proposition 56 of [59] that the $g_i(y)$'s that achieve the maximal information $I(U, Y)$ (where U are the features learned by $\mathbf{f}$ on X) are given by:

$$\mathbf{g}^*(y) = \mathrm{cov}(\mathbf{f}(X))^{-1}(\mathbb{E}_{p_{X|Y}(\cdot|y)}\left[\mathbf{f}(X)\right] - \mathbb{E}_{p_X}\left[\mathbf{f}(X)\right]),$$

$$\sigma = \mathbb{E}_{p_{X,Y}}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{g}(y)\right].$$

In this section, we will use these functions to perform soft feature selection and thus determine how best to combine these features to perform well on a new task given few training samples.

3.2.1 Combining Maximal Correlation Functions

Given a fixed set of feature functions $\{\mathbf{f}^{s_1}, \ldots, \mathbf{f}^{s_N}\}$, we seek to maximize the total maximal correlation

$$\mathcal{L} = \mathbb{E}_{\hat{P}^t_{X,Y}} \left[\mathbf{f}^{\mathrm{T}}(X)\, \mathbf{g}(Y)\right] \tag{3.1}$$

with respect to $\mathbf{g}$, where $\mathbf{f} = (\mathbf{f}^{s_1}, \ldots, \mathbf{f}^{s_N})^{\mathrm{T}}$ and $\mathbf{g} = (\mathbf{g}^{s_1}, \ldots, \mathbf{g}^{s_N})^{\mathrm{T}}$, and where the optimization is over all valid (zero-mean and unit-variance with respect to the empirical distribution of the target class labels) $\mathbf{g}$ for fixed $\mathbf{f}$. $\hat{P}^t_{X,Y}$ is the empirical joint target distribution of X and Y. We relax the orthogonality constraint here in order to simplify the expression and decouple the $g_i(y)$'s.

Expanding (3.1) as

$$\mathcal{L} = \sum_{i,n} \mathbb{E}_{\hat{P}^t_{X,Y}} \left[f_i^{s_n}(X)\, g_i^{s_n}(Y)\right], \tag{3.2}$$

we can then maximize each term separately, yielding

$$g_i^{s_n}(y) = \arg\max_{\tilde{g}_i^{s_n}} \mathcal{L} = \arg\max_{\tilde{g}_i^{s_n}} \mathbb{E}_{\hat{P}^t_{X,Y}} \left[f_i^{s_n}(X) \tilde{g}_i^{s_n}(Y)\right]. \tag{3.3}$$

Then, for each $g_i^{s_n}(y)$, for a fixed $f_i^{s_n}$, we have from (2.15) that the optimal $g_i^{s_n}$ is given by the conditional expectation

$$g_i^{s_n}(y) = \mathbb{E}_{\hat{P}^t_{X|Y}(\cdot|y)} \left[f_i^{s_n}(X)\right], \tag{3.4}$$

which can easily be computed from the target samples.

In turn, we compute the corresponding maximized correlation for each pair of functions $f_i^{s_n}$ and $g_i^{s_n}$ via

$$\sigma_{n,i} = \mathbb{E}_{\hat{P}^t_{X,Y}} \left[f_i^{s_n}(X)\, g_i^{s_n}(Y)\right]. \tag{3.5}$$

Algorithm 2 Extracting maximal correlation parameters

```
Data: Zero-mean, unit-variance feature functions $\{f_i^{s_n}\}$ from source tasks and target
      task samples $\{(x_1^t, y_1^t), \dots, (x_k^t, y_k^t)\}$
Result: Associated maximal correlations $\{\sigma_{n,i}\}$ and correlation functions $\{g_i^{s_n}\}$
for $n = 1, \dots, N$ do // Iterate over all source tasks
    for $i = 1, \dots, l_n$ do // Iterate over features in each network
        for $y \in \mathcal{Y}^t$ do // Iterate over all target class labels
            $g_i^{s_n}(y) \leftarrow \mathbb{E}_{P^t_{X|Y}(\cdot|y)}\left[f_i^{s_n}(X)\right]$ // Compute feature and label-specific
              weight
        end
        $\sigma_{n,i} \leftarrow \mathbb{E}_{\hat{P}^t_{X,Y}}\left[f_i^{s_n}(X)\, g_i^{s_n}(Y)\right]$ // Compute feature-specific weight
    end
end
return $\{g_i^{s_n}\}, \{\sigma_{n,i}\}$
```

3.2.2 The Maximal Correlation Weighting (MCW) Algorithm

Using the combining weights thus derived, a predictor for the target labels is formed in accordance with (2.6), specifically,

$$\hat{P}_{Y|X}(y|x) = \hat{P}_Y^t(y)\left(1 + \sum_{n,i} \sigma_{n,i} f_i^{s_n}(x) g_i^{s_n}(y)\right), \tag{3.6}$$

from which the prediction $\hat{y}$ for a given test sample x is

$$\hat{y} = \arg\max_y \hat{P}_{Y|X}(y|x) = \arg\max_y \hat{P}_Y^t(y)\left(1 + \sum_{n,i} \sigma_{n,i} f_i^{s_n}(x) g_i^{s_n}(y)\right), \tag{3.7}$$

where $\hat{P}_Y^t$ is an estimate of the target label distribution. The resulting algorithms for learning the MCW parameters and computing the MCW predictions are summarized in Algorithm 2 and Algorithm 3, respectively.

Computing the empirical conditional expected value requires a single pass through the data, and so has linear time complexity in the number of target samples. We also need to compute one conditional expectation for each feature function. Thus, the time complexity of the fine-tuning is $O(C + NKk)$, where C is the time needed to

Algorithm 3 Prediction with the maximal correlation weighting method

```
Data: Maximal correlation functions $\{f_i^{s_n}\}$ and $\{g_i^{s_n}\}$ with associated correlations
      $\{\sigma_{n,i}\}$, empirical class label distribution $\hat{P}_Y^t$, and target task sample $x^t$
Result: Class label prediction $\hat{y}^t$ given $x^t$
Initialize $\hat{P}_{Y|X}^t(y|x^t) = \hat{P}_Y^t(y)\ \forall y \in \mathcal{Y}^t$
for $n = 1, \ldots, N$ do // Iterate over all source tasks
    for $i = 1, \ldots, l_n$ do // Iterate over features in each network
        for $y \in \mathcal{Y}^t$ do // Iterate over all target class labels
            $\hat{P}_{Y|X}^t(y|x^t) = \hat{P}_{Y|X}^t(y|x^t) + \hat{P}_Y^t(y)\sigma_{n,i} f_i^{s_n}(x^t) g_i^{s_n}(y)$// Apply Equation (9)
        end
    end
end
return $\arg\max_y \hat{P}_{Y|X}^t(y|x)$
```

extract features from all the pre-trained networks, N is the number of networks, K is the maximum number of features per network, and k is the number of target training samples. The number of parameters grows as $O(NK|\mathcal{Y}^t|)$, which is the number of entries needed to store all the g functions. $|\mathcal{Y}^t|$ is the number of target class labels.

To compute a prediction from one target test sample, the time complexity is $O(C + NK|\mathcal{Y}^t|)$. This arises from the fact that we must compute the quantity $\sum_{n,i} \sigma_{n,i} f_i^{s_n}(x) g_i^{s_n}(y)$ for each possible class label.

3.3 Experimental Results

In order to illustrate the effectiveness of the MCW method, we perform experiments on three different image classification datasets: CIFAR-100, Stanford Dogs, and Tiny ImageNet. Example images from each dataset can be found in Figure 3-1.[1]

For each dataset, we divide the classes into a set of mutually exclusive subsets, select one subset as our target task, and several others as the source datasets. We use the LeNet architecture [71] as our neural network for each source dataset, and train a different network for each source dataset. We implemented the network in PyTorch

[1] Code for these experiments can be found at `https://github.com/jklee-mit/maximal_correlation_weighting`.

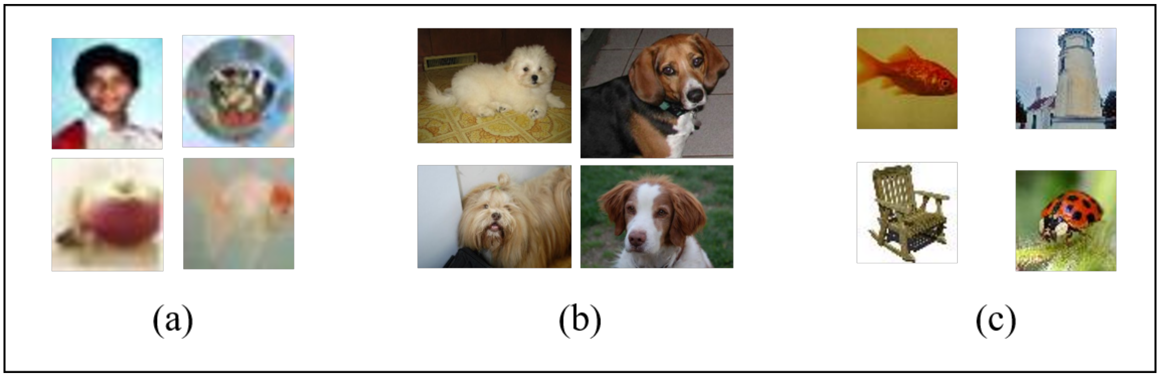


Figure 3-1: Example images from the (a) CIFAR-100, (b) Stanford Dogs, and (c) Tiny ImageNet datasets.

[96], and trained it for 100 epochs.

We remove the means and normalize to unit variance all of the feature functions with respect to the target samples, and then compute the maximal correlations and associated functions for each output in the penultimate layer using the target data according to Algorithm 2. We then use them to compute predictions on the test set for the target task according to Algorithm 3. The overall system is visualized in Figure 3-2.

We compare the classification accuracies on the test set with that of a support vector machine (SVM) trained on the penultimate layers with the same target training data (similar to the setup in [50]), as well as the best results from the MCW method and SVM method using only one source dataset/neural network. We also include the "upper bound" baseline performance on the dataset by a LeNet neural network trained on a number of target training samples equal to the number of training samples provided for each source dataset. The reported results are over 20 runs using the same set of tasks for each run.

3.3.1 CIFAR-100 Dataset

The CIFAR-100 dataset[2] [66] is a collection of color images of size 32x32 drawn from 100 different categories of real-world subjects. Because of the low resolution of the

[2] `https://www.cs.toronto.edu/~kriz/cifar.html`

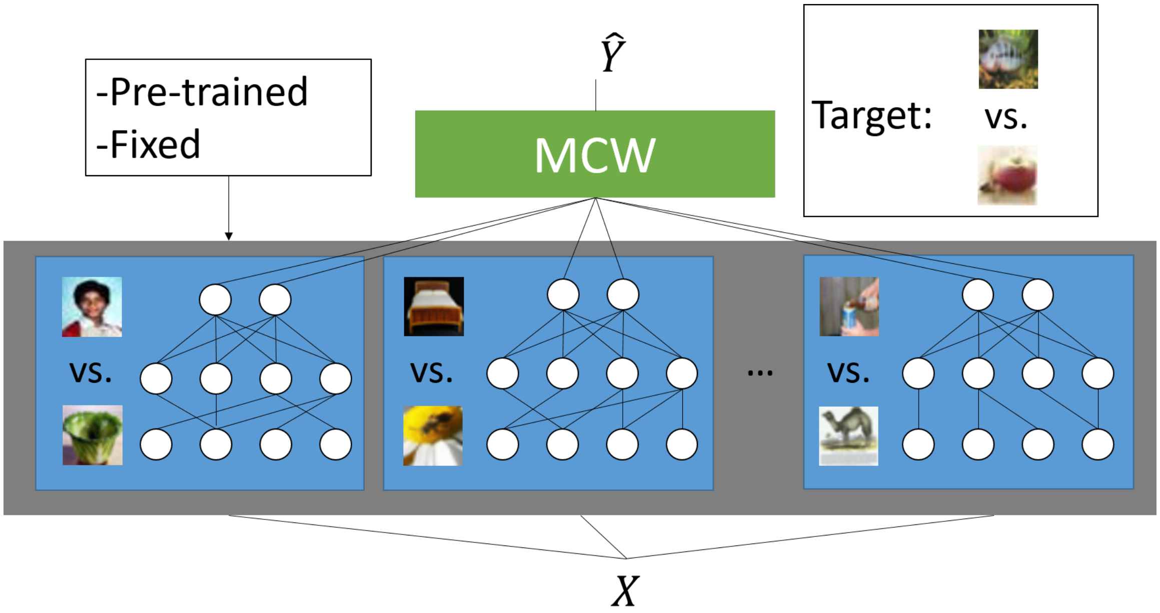


Figure 3-2: Block diagram illustrating the multiple source networks that are fed into the MCW algorithm to produce the final prediction.

images, CIFAR-100 is generally seen as a difficult classification problem. For our experiment, we construct a series of binary classification tasks from the classes. We randomly selected "apple" vs. "fish" as our target binary classification task, and randomly selected 10 other pairs of non-overlapping categories for the source tasks. For each source task, we extracted 500 source samples per class for training, and we used 1, 5, 10, and 20 target samples per class to compute the maximal correlation functions in the target task. We used the training/test splits included with the dataset, and report results over all test samples with the target labels.

Table 3.1 shows the test accuracies of our algorithm as applied to the CIFAR-100 dataset. We can see that the MCW method performs significantly better than an SVM when there are few samples, likely due to its ability to work with fewer target data points in learning, but that this performance gap closes as more target training samples are added, likely due to the fact that the models that require joint training over the features begin to have enough target samples to properly learn their parameters. In addition, we can see that combining multiple networks provides performance that is better than any one network can achieve with the same methods,

Table 3.1: Experimental results for the CIFAR-100 dataset. Accuracies are reported with 95% confidence intervals.

Method	1-Shot Acc.	5-Shot Acc.	10-Shot Acc.	20-Shot Acc.
Best 1-Source SVM	56.9 ± 2.5	67.0 ± 3.0	70.4 ± 1.9	70.9 ± 1.2
Best 1-Source MCW	59.2 ± 2.1	69.0 ± 3.0	67.0 ± 2.4	70.4 ± 1.5
Multi-Source SVM	64.7 ± 3.0	72.8 ± 2.7	76.2 ± 1.8	81.5 ± 0.6
Multi-Source MCW	**69.0 ± 3.0**	**78.1 ± 0.8**	**80.1 ± 0.8**	**81.7 ± 0.6**
Baseline (All Target Samples)	90.7 ± 0.1			

once again suggesting that our algorithm is taking in contributions from multiple sources instead of just one.

In order to investigate the functioning of the MCW method for feature selection, we plot the sum of correlations for each of the 10 tasks for the 5-shot case in Figure 2. We can see a significant variation among tasks, which provides a clear indication of which tasks are being preferred and which do not contribute as much to the overall performance. To verify this, we run two additional experiments in which we first remove the source task with the lowest total correlation ("camel" vs. "can") and see how well the MCW method performs with the remaining 9 source datasets, and then remove the task with the highest total correlation ("dolphin" vs. "elephant") while keeping the other 9 sources in and run the same test.

Without the least-favoured task, the classification accuracy drops to 76.8 ± 1.0, which is not a significant difference from using all 10 source tasks. However, when we remove the most-favoured task, the accuracy plummets to 73.0 ± 1.3, which indicates that "dolphin" vs. "elephant" had a significant impact on the quality of the classifier, but that the MCW method still takes the input of the other tasks into account in order to construct a good classifier on the target set.

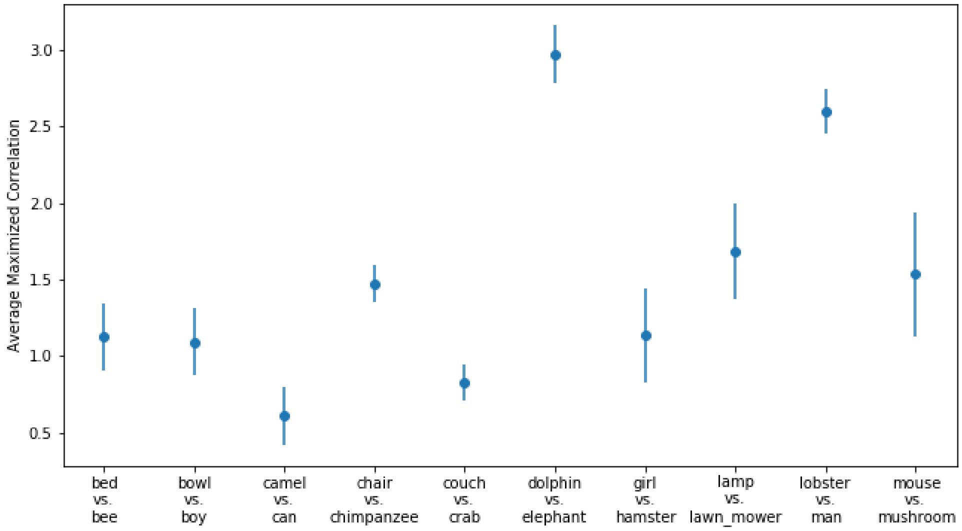


Figure 3-3: Average values of $\sum_i \sigma_{n,i}$ for each source task s_n for the 5-shot transfer learning task on the CIFAR-100 dataset, with the target task of "apple vs. fish." Points are plotted with 95% confidence intervals.

3.3.2 Stanford Dogs Dataset

The Stanford Dogs dataset[3] [63] is a subset of the ImageNet dataset designed for fine-grained image classification. It consists of 22,000 images of varying sizes covering 120 classes of dog breeds. For this task we construct a random 5-way target classification task (differentiating between "Chihuahua", "Japanese Spaniel", "Maltese Dog", "Pekinese", and "Shih-Tzu") and 10 other random 5-way source classification tasks with no overlapping classes. For the target set, we take 5 samples per class for training and use the rest for testing. For the source sets, we take 100 samples per class for training. All images were resized to size 144x144.

Table 3.2 shows the test accuracies of our algorithm as applied to the Stanford Dogs dataset. This time, we observe a loose hierarchy whereby the MCW method outperforms the SVM, which in turn outperforms any single source transfer. We can thus conclude that the MCW method is effective in the case of m-way learning for $m > 2$, and that we can still leverage multiple networks to get a gain in cases where

[3] `http://vision.stanford.edu/aditya86/ImageNetDogs/`

Table 3.2: Experimental results for the Stanford Dogs dataset. Accuracies are reported with 95% confidence intervals.

Method	5-Shot Accuracy
Best Single Source SVM	35.8 ± 0.8
Best Single Source MCW	38.2 ± 0.6
Multi-Source SVM	38.9 ± 0.3
Multi-Source MCW	**41.6 ± 0.5**
Baseline (All Target Samples)	55.2 ± 0.1

the classes are very similar.

3.3.3 Tiny ImageNet Dataset

The Tiny ImageNet dataset[4] [74] is another subset of the ImageNet dataset, consisting of images of size 64x64 drawn from 200 categories, with 500 images provided for each category. The categories cover a much wider range than the Stanford Dogs dataset, including animals, natural and man-made objects, and even abstract concepts (e.g., "elongation"). As with the Stanford Dogs dataset, we constructed 11 random 5-way classification tasks, and selected one as the target task ("Lighthouse" vs. "Rocking Chair" vs. "Bannister" vs. "Jellyfish" vs. "Chain") and the others as source tasks. We used 5 training samples per class for the target task (with 250 samples per class for testing) and all 500 samples per class for the source training samples. For the baseline, we only trained with the 250 samples per class in the target dataset that were not in the test split.

Table 3.3 shows the test accuracies of the MCW method as applied to the Tiny ImageNet dataset. Compared to the Stanford Dogs dataset, we see a larger gain from leveraging multiple sources compared to a single source, which suggests that if the source classes are much more dissimilar than the target classes, then integrating more networks (and thus leveraging a wider range of features) will have a greater effect on target task accuracy, likely due to the ability of different source tasks to "cover" the feature set needed for the target task, as opposed to the Dogs setup where the classes

[4] `https://tiny-imagenet.herokuapp.com/`

Table 3.3: Experimental results for the Tiny ImageNet dataset. Accuracies are reported with 95% confidence intervals.

Method	5-Shot Accuracy
Best Single Source SVM	31.4 ± 0.9
Best Single Source MCW	33.9 ± 1.0
Multi-Source SVM	42.5 ± 1.4
Multi-Source MCW	$\mathbf{47.4 \pm 1.1}$
Baseline (All Target Samples)	53.8 ± 0.1

were highly similar.

3.4 Concluding Remarks

In this chapter, we showed that the HGR can also be used to perform feature selection, in both a soft manner (weighting) as well as a hard manner (removing unimportant features entirely). Furthermore, we see in this multi-source problem the interpretive power of the HGR, as the maximal correlations can be used to evaluate the extent to which each feature is used to make the final prediction, and thus decide which subset of features would be most useful to keep if there are limitations as to how many can be used.

A less-decoupled perspective may prove useful for better evaluating the effectiveness of subsets of features, allowing for more effective feature selection in the future. In addition, the privacy implications of our setup could be considered, as it is possible to reconstruct training data from the learned features [38], which means that our method as-is does not erase all privacy concerns. These methods can be countered with differential privacy measures [85], such as adding noise to the feature functions, but their effect on transfer quality is as-of-yet unknown.

Chapter 4

Sensitivity-Aware Learning Algorithms

In Chapter 3, we illustrated the use of the HGR in feature selection. In this chapter, we will consider the problem of feature suppression instead. In particular, we will look at problems involving enforcing sensitivity criteria in learning. These problems are widely applicable to multiple fields, including fair machine learning, privacy, and transfer learning (specifically, domain generalization), and thus having a single paradigm to solve them provides an opportunity to glimpse the range and ease of adaptability of our method. As well, this is a good illustration of both the power as well as the limitations of an HGR-based approach to solving these problems.

In general, these problems assume a primary learning objective defined by predicting a target Y from data X, such as classification or regression, measured using some loss function $L(\hat{Y}, Y)$ (e.g., cross-entropy $\mathbb{E}_{P_Y}\left[\log P_{\hat{Y}(Y)}\right]$ for classification and mean-squared error $\mathbb{E}\left[(\hat{Y} - Y)^2\right]$ for regression). They also assume the existence of a third variable, $D \in \mathcal{D}$, which defines a secondary objective.

In these cases, we have access to samples $\{(x_1, y_1, d_1), \ldots, (x_n, y_n, d_n)\}$ from which to learn the model. While we assume in this case that we always have access to all three variables during training, in some cases, our model may only be allowed to make inferences using X during test time, rather than on both X and D.

The secondary objectives we are interested in are independence-based sensitivity criteria, which demand some kind of marginal or conditional independence between the learned features $\Theta(X)$ or predictions $\hat{Y}$ and the variables D and/or Y. These criteria can be used to model a wide range of real-world problems, from fair machine learning to privacy to domain generalization.

4.1 Independence, Separation, and Sufficiency

We look at three sensitivity criteria, which cover the set of possible non-trivial marginal and conditional independence criteria one could have between the variables in question. Since all three of these criteria are used in fair machine learning, we will be using the terms from this field to refer to them.

The first criterion is known as *independence*, and states that any prediction we make must be (marginally) independent of the sensitive attribute, that is,[1] $\hat{Y} \perp D$.

The purpose of this criterion is very simple; it is used to suppress any information about D from the prediction alone. For example, if D represents information that one wishes to remain private, then independence can be used to ensure that the predictions do not leak this private information [18]. In fair machine learning, this criterion can also be used enforce equal treatment or allocation of resources among individuals with a shared identity, such as race or sex, also known as enforcing *demographic parity* [8].

In some cases, such equal allocation via independence between the prediction and the sensitive attribute is not the goal. Instead, we may desire to have equal levels of performance across subsets of the population, for example, equal accuracy $P(\hat{Y} = Y)$ or equal recall $P(\hat{Y} = Y|Y = k)$ across all relevant subsets. The second criterion we introduce is *separation*, which enforces this behaviour. Separation states that any prediction we make must be independent of the sensitive attribute, conditioned on the true label, that is, $\hat{Y} \perp D|Y$. In fair machine learning, this allows us to enforce the condition of *equalized opportunities*, which requires that the same recall are achieved among all relevant subsets of the population [8].

Our final criterion is *sufficiency*, which states that our prediction must contain all

[1] We note here that $\perp$ is used exclusively to denote independence of random variables.

Table 4.1: Types of sensitivity criteria.

Independence	Separation	Sufficiency
$\hat{Y} \perp D$	$\hat{Y} \perp D \vert Y$	$Y \perp D \vert \hat{Y}$

the information about the sensitive attribute needed to predict the target, that is, $Y \perp D|\hat{Y}$. This criterion is useful for ensuring positive and negative predictive parity (e.g., equal precision $P(\hat{Y} = Y|\hat{Y} = k)$ across all groups) [8], and has applications in both domain generalization [5] and fair selective classification [62]. Table 4.1 summarizes the three criteria.

At this point, we make a few remarks about these criteria. First of all, these criteria are largely mutually exclusive outside of some trivial cases. Specifically, we have the following results:

Proposition 1. *If Y and D are not independent, then sufficiency and independence cannot both hold. [8]*

Proposition 2. *If Y and D are not independent, and if all events in the joint distribution of (X, Y, D) have positive probability, then sufficiency and separation cannot both hold. [8, 122]*

Proposition 3. *If Y and D are not independent, $\hat{Y}$ and D are not independent, and Y is binary, then separation and independence cannot both hold. [8]*

In addition, in many applications, we do not require that these criteria be perfectly satisfied, and instead formulate measures of violation to quantify deviations from perfect satisfaction of the criteria. Then, we can evaluate algorithms by looking at the tradeoff between severity of violation and overall performance on the primary objective.

4.2 Enforcing Sensitivity Criteria: Discrete Case

When X, Y, and D are all discrete, we look to the matrix-based representations of the probability distributions in order to derive features that balance the primary learning objective with the sensitivity criteria.

In all cases, we assume that we wish to learn a feature mapping $\mathbf{f} \in \mathbb{R}^{|\mathcal{X}| \times k}$ from X with respect to Y that maximizes the maximal correlation under the variational characterization given in Section 2.5.

4.2.1 Independence

When we apply the independence criterion, we obtain the following constrained optimization problem:

$$\begin{aligned} \max_{\substack{\mathbf{f}\colon \mathcal{X} \to \mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad & \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] \\ \text{s.t.} \quad & U \perp D, \end{aligned} \tag{4.1}$$

where U denotes the features produced by $\mathbf{f}$ on X. We can relax this constraint to produce

$$\begin{aligned} \max_{\substack{\mathbf{f}\colon \mathcal{X} \to \mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad & \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] \\ \text{s.t.} \quad & \mathbb{E}_{P_D}\left[\|\mathbb{E}_{P_{X|D}}\left[\mathbf{f}(X)\right]\|^2\right] = 0. \end{aligned} \tag{4.2}$$

Transforming this constraint into a penalty, we obtain

$$\max_{\substack{\mathbf{f}\colon \mathcal{X} \to \mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] - \lambda \mathbb{E}_{P_D}\left[\|\mathbb{E}_{P_{X|D}}\left[\mathbf{f}(X)\right]\|^2\right]. \tag{4.3}$$

where λ is a regularization parameter. Rewriting this in terms of the DTM yields

$$\max_{\mathbf{\Phi}_X^{\mathrm{T}}\mathbf{\Phi}_X=\mathbf{I}} \operatorname{tr}(\mathbf{\Phi}_X^{\mathrm{T}}\big(\mathbf{B}_{X,Y}^{\mathrm{T}}\mathbf{B}_{X,Y} - \lambda \mathbf{B}_{X,D}^{\mathrm{T}}\mathbf{B}_{X,D}\big)\mathbf{\Phi}_X). \tag{4.4}$$

Then, solving this optimization is equivalent to finding the eigendecomposition of $\mathbf{B}_{X,Y}^{\mathrm{T}}\mathbf{B}_{X,Y} - \lambda \mathbf{B}_{X,D}^{\mathrm{T}}\mathbf{B}_{X,D}$ [116]. When we are given samples from $P_{X,Y,D}$ from which to learn the features, we can perform the decomposition using the empirical forms of $\mathbf{B}_{X,Y}$, which we denote as $\hat{\mathbf{B}}_{X,Y}$.

4.2.2 Separation

The separation criterion leads us to a similar constrained optimization problem as in independence:

$$\begin{aligned} \max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad & \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] \\ \text{s.t.} \quad & U \perp D|Y. \end{aligned} \tag{4.5}$$

We can replace the constraint with an equivalent one using the mutual information:

$$\begin{aligned} \max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad & \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] \\ \text{s.t.} \quad & I(U;D|Y) = 0, \end{aligned} \tag{4.6}$$

which can be transformed into the relaxed problem

$$\max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] - \lambda I(U;D|Y). \tag{4.7}$$

At this point, we can make use of the chain rule of mutual information to obtain:

$$\max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] - \lambda\big(I(U;(D,Y)) - I(U;Y)\big). \tag{4.8}$$

Since we are minimizing $I(U;(D,Y))$, we will be trying to learn features that are weakly dependent with respect to (D,Y), in which case, we can apply the HGR approximation of the mutual information to obtain

$$\begin{aligned} \max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad & \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] \\ & - \lambda\big(\mathbb{E}_{P_{D,Y}}\left[\|\mathbb{E}_{P_{X|D,Y}}\left[\mathbf{f}(X)\right]\|^2\right] - \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right]\big). \end{aligned} \tag{4.9}$$

This simplifies to

$$\max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} (1+\lambda)\mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] - \lambda\mathbb{E}_{P_{D,Y}}\left[\|\mathbb{E}_{P_{X|D,Y}}\left[\mathbf{f}(X)\right]\|^2\right], \tag{4.10}$$

and can thus be solved similarly via the eigendecomposition of $(1+\lambda)\mathbf{B}_{X,Y}^{\mathrm{T}}\mathbf{B}_{X,Y} - \lambda\mathbf{B}_{X,D\times Y}^{\mathrm{T}}\mathbf{B}_{X,D\times Y}$, where $\mathbf{B}_{X,D\times Y}$ denotes the DTM of the joint distribution between X and the Cartesian product of D and Y.

4.2.3 Sufficiency

Our final criteria, sufficiency, is more difficult to analyze using the HGR. We can write the constrained problem as before,

$$\begin{aligned} \max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad & \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] \\ \text{s.t.} \quad & Y \perp D|U, \end{aligned} \tag{4.11}$$

and then apply the mutual information-based relaxation to obtain

$$\max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0} \\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] - \lambda I(Y;D|U). \tag{4.12}$$

We can apply the chain rule to $I(Y;D|U)$ as follows:

$$\begin{aligned} & I(Y;D|U) \\ =& I((U,D);Y) - I(U;Y) \\ =& I(U;Y|D) + I(Y;D) - I(U;Y) \\ =& I(U;(Y,D)) - I(U;D) + I(Y;D) - I(U;Y), \end{aligned} \tag{4.13}$$

Since $I(Y;D)$ is constant for all choices of $\mathbf{f}$, we can ignore it when we plug in the above expression into the optimization, which yields

$$\max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k\\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0}\\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad \mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] - \lambda\big(I(U;(Y,D)) - I(U;D) - I(U;Y)\big). \tag{4.14}$$

If we apply the usual approximation of mutual information using HGR, we can obtain the following optimization:

$$\begin{aligned}\max_{\substack{\mathbf{f}\colon \mathcal{X}\to\mathbb{R}^k\\ \mathbb{E}[\mathbf{f}(X)]=\mathbf{0}\\ \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(X)\mathbf{f}(X)\right]=\mathbf{I}}} \quad & (1+\lambda)\mathbb{E}_{P_Y}\left[\|\mathbb{E}_{P_{X|Y}}\left[\mathbf{f}(X)\right]\|^2\right] - \lambda\Big(\mathbb{E}_{P_{D,Y}}\left[\|\mathbb{E}_{P_{X|D,Y}}\left[\mathbf{f}(X)\right]\|^2\right] \\ & \qquad - \mathbb{E}_{P_D}\left[\|\mathbb{E}_{P_{X|D}}\left[\mathbf{f}(X)\right]\|^2\right]\Big).\end{aligned} \tag{4.15}$$

Once again, we can then compute the features by solving the eigendecomposition of $(1+\lambda)\mathbf{B}_{X,Y}^{\mathrm{T}}\mathbf{B}_{X,Y} - \lambda\big(\mathbf{B}_{X,D\times Y}^{\mathrm{T}}\mathbf{B}_{X,D\times Y} - \mathbf{B}_{X,D}^{\mathrm{T}}\mathbf{B}_{X,D}\big)$.

However, we note that since we are trying to maximize $I(U;D)$, it is unlikely that we will be in the weakly dependent regime, and thus the approximation is not guaranteed to be close enough to be useful in this case. This is especially problematic as we are trying to minimize a difference between two very similar quantities, and thus the effects of a poor approximation can be exacerbated and result in the wrong features being learned. Still, intuitively, we would expect this objective to produce sufficient features by penalizing the dependence between the features and the joint distribution of Y and D while maximizing the individual dependencies between the features and each of Y and D.

4.3 Enforcing Sensitivity Criteria: Continuous Case

In the continuous case, we once again assume some primary loss function $L(\hat{Y},Y)$ for which we seek a set of features $\Theta(X)$ that both satisfy some sensitivity criteria (or minimize the penalty based on it) while still allowing for a good predictor $\hat{Y} = T(\Theta(X))$ to be learned that minimizes the primary loss function. For this section, we will assume that $\Theta(X) \in \mathbb{R}^m$.

When X, Y, and D are all continuous and real-valued, however, computing the HGR maximal correlation becomes much more difficult, since we are searching the space of functions over real numbers. We thus turn to approximations, and begin by limiting our scope of learning algorithms to those which train models (e.g., neural nets) via (stochastic) gradient descent using samples, which encompasses the applications of interest to us.

It then follows that any approximation of the HGR maximal correlation used must be differentiable to make it possible to calculate the gradient. We thus restrict the space of maximal correlation functions to be the family of functions that can be learned by neural nets, allowing us to compute the gradient while still providing a rich set of functions to search over.

4.3.1 Independence

In the continuous case, our constrained optimization problem for independence looks like

$$\begin{aligned} \min_{\Phi, T} \quad & L(\hat{Y}, Y) \\ \text{s.t.} \quad & U \perp D, \end{aligned} \tag{4.16}$$

where U denotes the features produced by Θ on X. We can rewrite this using the HGR as

$$\begin{aligned} \min_{\Theta, T} \quad & L(\hat{Y}, Y) \\ \text{s.t.} \quad & \text{HGR}(U; D) = 0. \end{aligned} \tag{4.17}$$

The relaxed form of this is

$$\min_{\substack{\Theta\colon \mathcal{X} \to \mathbb{R}^m \\ T\colon \mathbb{R}^m \to \mathcal{Y}}} L(T(\Theta(X)), Y) + \lambda \text{HGR}(U; D). \tag{4.18}$$

Given the difficulty of enforcing the orthogonalization constraint, we use the soft-

HGR discussed in Section 2.6.2. Then, our learning objective becomes

$$\min_{\substack{\Theta\colon \mathcal{X}\to\mathbb{R}^m \\ T\colon \mathbb{R}^m\to\mathcal{Y}}} \max_{\substack{\mathbf{f}\colon \mathbb{R}^m\to\mathbb{R}^k,\ \mathbf{g}\colon \mathcal{D}\to\mathbb{R}^k \\ \mathbb{E}[\mathbf{f}(\Theta(X))]=\mathbb{E}[\mathbf{g}(D)]=\mathbf{0}}} C, \tag{4.19}$$

where

$$\begin{aligned} C =& L(T(\Theta(X)), Y) + \lambda \mathbb{E}\left[\mathbf{f}^{\mathrm{T}}(\Theta(X))\,\mathbf{g}(D)\right] \\ & - \frac{\lambda}{2}\operatorname{tr}\big(\operatorname{cov}[\mathbf{f}(\Theta(X))]\operatorname{cov}[\mathbf{g}(D)]\big). \end{aligned}$$

We solve this by alternating between optimizing Θ, T and optimizing $\mathbf{f}, \mathbf{g}$.

4.3.2 Separation

For separation, we use a similar argument as in the discrete case to ensure the conditional independence. Our constrained optimization for continuous separation is:

$$\begin{aligned} \min_{\Theta,T} \quad & L(\hat{Y}, Y) \\ \text{s.t.} \quad & U \perp D|Y, \end{aligned} \tag{4.20}$$

which we can rewrite using the mutual information as:

$$\begin{aligned} \min_{\Theta,T} \quad & L(\hat{Y}, Y) \\ \text{s.t.} \quad & I(U; D|Y) = 0. \end{aligned} \tag{4.21}$$

The relaxed form of this is

$$\min_{\substack{\Theta\colon \mathcal{X}\to\mathbb{R}^m \\ T\colon \mathbb{R}^m\to\mathcal{Y}}} L(T(\Theta(X)), Y) + \lambda I(U; D|Y). \tag{4.22}$$

Then, by applying the chain rule and the same HGR approximation of the mutual information, then converting HGR to soft-HGR, we obtain:

$$\begin{aligned}\min_{\substack{\Theta\colon \mathcal{X}\to\mathbb{R}^m\\ T\colon \mathbb{R}^m\to\mathcal{Y}}} & L(T(\Theta(X)),Y)+ \\ & \lambda\Big(\mathrm{HGR}_{\text{soft}}(U;D\times Y)-\mathrm{HGR}_{\text{soft}}(U;Y)\Big).\end{aligned} \tag{4.23}$$

Note that for the first soft-HGR term, we use $\mathbf{g}, \mathbf{h}$ to denote the maximal correlation functions, and $\mathbf{g}', \mathbf{h}'$ to denote the functions for the second term. Similar to the discrete case, the difference term allows us to approximate the conditional mutual information using two unconditional terms. Once again, we solve this optimization by alternating between optimizing $\mathbf{f}$, T and optimizing $\mathbf{g}, \mathbf{h}, \mathbf{g}', \mathbf{h}'$.

4.3.3 Sufficiency

For sufficiency, we encounter the same issue as in the discrete case. We have the constrained optimization

$$\begin{aligned}\min_{\Theta,T} \quad & L(\hat{Y},Y) \\ \text{s.t.} \quad & Y \perp D|U,\end{aligned} \tag{4.24}$$

which we can rewrite using the mutual information as

$$\begin{aligned}\min_{\Theta,T} \quad & L(\hat{Y},Y) \\ \text{s.t.} \quad & I(Y;D|U)) = 0.\end{aligned} \tag{4.25}$$

The relaxed form of this is

$$\min_{\substack{\Theta\colon \mathcal{X}\to\mathbb{R}^m\\ T\colon \mathbb{R}^m\to\mathcal{Y}}} L(T(\Theta(X)),Y)+\lambda I(Y;D|U). \tag{4.26}$$

Using the same chain rule manipulations and HGR substitutions, we can obtain

$$\begin{aligned}\min_{\substack{\Theta\colon \mathcal{X}\to\mathbb{R}^m\\ T\colon \mathbb{R}^m\to\mathcal{Y}}} \ & L(T(\Theta(X)), Y) + \\ & \lambda\big(\mathrm{HGR}_{\mathrm{soft}}(U; D \times Y) - \mathrm{HGR}_{\mathrm{soft}}(U; Y) - \mathrm{HGR}_{\mathrm{soft}}(U; D)\big).\end{aligned} \tag{4.27}$$

However, once again, we cannot guarantee that we are operating in the weakly-dependent regime, and thus cannot guarantee that this will be a good approximation to use for this optimization, though we would expect similar behaviour as in the discrete case in enforcing sufficiency.

4.4 Concluding Remarks

In this section, we have outlined the methods by which one can enforce sensitivity criteria in different modalities of data. In doing so, we also exposed a possible weakness of the HGR-based approach to enforcing sufficiency.

Finding a different way to formulate the objective that does not suffer from the same instability would be a highly beneficial next step. However, it may not be possible to construct a set of sufficient features from X if Y and D are dependent in some way that does not involve X at all. For example, if Y and D are both correlated with some variable Z that is completely independent from X, then sufficiency can never be achieved by any $\Theta(X)$.

In the next chapter, we will explore the space of problems that can be solved by enforcing sensitivity criteria, to illustrate both the effectiveness of our method empirically, as well as to provide some insights on the many different ways these criteria can be used to model and solve real-world problems.

Chapter 5

Applications of Sensitivity-Aware Learning

A tool is only as useful as the problems it can solve. Equipped with a suite of methods for solving machine learning problems with sensitivity criteria, we now look at a number of different applications for these methods to illustrate the breadth of problems that can be tackled using this paradigm, and how even some problems that may not appear to be applicable to this paradigm at first can still be formulated in such a way to allow us to use our tool effectively.

We will begin with a look at fair machine learning, a field of vital importance that contains both obvious and non-obvious use cases for sensitivity criteria. We then turn our attention to privacy, a problem that mirrors the problem of demographic parity in fair machine learning, and show how this connection between the two allows us to apply our methods with only minor modifications. Finally, we jump far afield to a problem in transfer learning, where a completely different setting (domain generalization) turns out to be linked back to previous problems via the sufficiency criterion.

5.1 An Overview of Fair Machine Learning

As the scope and diversity of machine learning applications continue to grow, the use of learning algorithms in many industries have raised a number of ethical and legal concerns. One especially important concern is that of fairness and bias in predictions made by automatic systems [109, 9]. With these systems being trusted to aid or make decisions regarding loan applications, criminal sentencing, and even health care, it is vital to ensure that unfair biases cannot influence them.

In general, fairness issues arise when a system wishes to make a prediction or decision based on some set of given attributes, but such decisions should not lead to a systematic disadvantage in outcomes for groups or individuals based on some sensitive attribute about which we desire to be "fair". For example, in the criminal justice system, one might wish to make predictions about the chance of recidivism of a convicted criminal given factors such as the nature of the crime and the number of prior arrests, but such predictions should not be determined by race (the sensitive attribute). This is a known issue with the COMPAS recidivism score, which, despite not using race as an input to make decisions, still leads to systematic bias towards members of certain races in the output score [3, 25].

In the case of fair machine learning, all three sensitivity criteria have very clear interpretations in terms of implementing different types of fairness. Additionally, for this chapter, when D is discrete, we also refer to members sharing the same value of D as being part of the same *group*. For example, this could represent people of the same gender or the same ethnic background.

5.2 Demographic Parity and Equalized Opportunities

Demographic parity and equalized opportunities are two very standard definitions of fairness, each of which have their own advantages and disadvantages.

Demographic parity requires the distribution of predictions/decisions to be independent of the sensitive attribute, that is, $\hat{Y} \perp D$. This is exactly the independence criterion, and from a fairness perspective, is a very intuitive criterion, as it forces the same distribution of outcomes between different groups [8].

One classic example of this criterion being used is in affirmative action, which requires that a company hire the same proportion of candidates from each group (in this case, D is discrete), that is, it requires that $P(\hat{Y}|D = a) = P(\hat{Y}|D = b) \quad \forall a, b \in \mathcal{D}$, which can be satisfied by imposing independence/demographic parity. In this particular case, the US Equal Employment Opportunity Commission [26] also provides a measure for unfairness known as *disparate impact*, which is defined as $D(P(\hat{Y}|D = 1); P(\hat{Y}|D = 0)) = \frac{P(\hat{Y}=1|D=0)}{P(\hat{Y}=1|D=1)}$.

Equalized opportunities requires the distribution of predictions/decisions to be independent of the sensitive attribute, conditioned on the true labels, that is, $\hat{Y} \perp D|Y$. This corresponds to the separation criterion, and from a fairness perspective, ensures parity of predictive power between groups [8].

In particular, we also have that the false negative rates between groups must be equal, as well as the false positive rates, since equalized opportunities implies that $P(\hat{Y}|Y, D = a) = P(\hat{Y}|Y, D = b) \quad \forall a, b \in \mathcal{D}$. This leads us to a standard measure for unfairness in the case of equalized opportunities when D is binary, known as difference in equalized opportunities (DEO):

$$\text{DEO} = P(\hat{Y} = 1|D = 1, Y = 1) - P(\hat{Y} = 1|D = 0, Y = 1) \tag{5.1}$$

Both measures have their benefits and drawbacks, and unfortunately, are impossible to satisfy at the same time. Demographic parity is very useful in ensuring the fair allocation of resources [23], but in cases where the marginal distribution of labels are different for different groups, this same condition results in incorrect allocation of labels, which may result in the perception that certain groups are favoured unfairly by the system.

Meanwhile, equalized opportunities ensures that no group is especially penalized by a poorly-performing classifier, and is especially important in applications such as facial recognition, where it has been shown that certain face classifiers perform better on some racial groups than others [124, 1, 120]. In more unfortunate cases, Twitter's smart cropping system has also been accused of selecting certain racial groups over others when trying to determine what part of an image is "important" [126], and juries that are presented with forensic evidence may be given incorrect statistics on its accuracy due to group biases, resulting in an improper weighting of said evidence [27]. Equalized opportunities also arises when designing systems for loan allocations [125] and recidivism prediction [3], as we desire these systems to perform equally well on all groups, as a difference in accuracy could result in one group being seen as more likely to default on loans or commit crimes due to incorrect decisions being made [118]. However, this criterion has been criticized for poorly handling the tails of population distributions [28].

Thus, we desire a framework that is flexible enough to handle different fairness criteria, and to do it with different modalities of data (e.g., discrete vs. continuous data). The HGR-based method for enforcing these criteria are thus well-suited for solving such problems.

This bias mitigation must also be balanced out with the system's usefulness, and often one must tune the tradeoff between the fairness (as measured in the particular context) and performance, which can be a difficult process if the fairness-performance curve is not smooth. Generating the frontier of possible values can be computationally infeasible or impossible if the algorithm does not have a regularization parameter to adjust (see [19, 86]), thus making it difficult to achieve this balance, which makes fast generation of fair classifiers even more important. Once again, the speed at which we can compute the HGR serves us well here.

5.2.1 Prior Work

Independence and separation in fairness have been studied in many works. Most existing approaches fail to provide an efficient solution in both discrete/continuous settings for both independence and separation.

Zemel et al. [132], Hardt et al. [53], and Calmon et al. [19] require D to be discrete, and only work on either independence [132, 19] or separation [53]. Calmon et al. [19] also requires pre-processing the entire dataset, which is computationally expensive.

Other methods can also be limited in their ability to handle all dependencies between variables. Zafar et al. [131] uses a covariance-based constraint to enforce fairness, so it likely would not do well on other metrics. Furthermore, it is strictly a linear penalty rather than our non-linear formulation and penalizes the predictions of the system rather than the features learned. This limits the relationships between variables it can capture. Adversarial methods are proposed in [133, 2] to enforce independence or separation, but requires the training of an adversary to predict the sensitive attribute, which can introduce issues of convergence and bias, and does not include a specific treatment for the discrete case.

Recently, Mary et al. [86] propose the use of the HGR maximal correlation as a regularizer for either the independence or the separation constraint. In contrast to our approach dealing with the maximal correlation directly, they use a chi-squared divergence computed over a mesh grid to upper bound the HGR maximal correlation during the optimization of the classifier (either a linear regressor or a deep neural net (DNN)). This method applies to cases where X is continuous and Y and D are either continuous or discrete variables, but scales poorly with the bandwidth and dimensionality of D, and treats the discrete case in the same way as the continuous case, resulting in slow performance on discrete datasets.

There are other works that use either an HGR-based or mutual information-based formulation of fairness, but do not generalize to more than one setting. Grari et al. [46], Moyer et al. [88], and Baharlouei et al. [6] use correlation-based regularizers, but only for the independence case, and their methods are not designed for continuous sensitive attributes. Finally, Cho et al. [24] approximates the mutual information with a variational formulation, but does not include a formulation for continuous labels.

5.2.2 Experimental Setup: Discrete Case

In order to illustrate the applications of our regularizers for demographic parity and equalized opportunities, we test them on standard fairness datasets, both those containing discrete data and those containing continuous data, and look at how the frontier of possible values for performance vs. fairness compare with other methods.

For the discrete case, we test the proposed DTM-based approach on ProPublica's COMPAS recidivism dataset[1] and the UCI Adult dataset[2], which were chosen as they contain categorical features and are used in prior works.

For the COMPAS dataset [100], the goal is to predict whether the individual recidivated (re-offended) Y using the severity of charge, number of prior crimes, and age category as the decision variables X. As discussed in [19], COMPAS scores are biased against African-Americans, so race is set to be the sensitive attribute D and filtered to contain only Caucasian and African-American individuals. The filtered dataset contains 6172 samples.

The Adult dataset [65] consists of census data drawn from the 1994 Census database, with 48,842 samples. The goal is to predict the binary indicator Y of whether the income of the individual is more than 50K or not based on the following decision variables X: age (quantized to decades) and education (in years), and the sensitive attribute D is the sex of the individual.

For both datasets, we randomly split all data into 80%/20% training/test samples. We first construct an estimate of the relevant DTMs with the empirical distribution of the training set, then solve the proposed optimizations in Sections 4.2.1 and 4.2.2 to obtain fair feature mappings $\hat{\mathbf{f}}(x), \hat{\mathbf{g}}(y)$. The predictions $\hat{Y}$ of the test samples X' are given by plugging the learned feature mappings $\hat{\mathbf{f}}(x'), \hat{\mathbf{g}}(y)$ into the MAP rule (2.6), where P_Y can be estimated from the empirical distribution $\hat{P}_Y$ on the training set. We sweep over values of λ from 0 to 2.5 in the demographic parity case, and from 0 to 1 in the equalized opportunities case.

For the demographic parity case, we compare the trade-off between the performance and the discrimination achieved by our method with that of the optimized pre-processing methods proposed in [19]. Note that we adopt the same settings as the experiments in [19] to do a fair comparison, and the reported results for their

[1] `https://github.com/propublica/compas-analysis`

[2] `https://archive.ics.uci.edu/ml/datasets/adult`

method are from their work. We plot the area under receiver operating characteristic curve (AUC) of $\hat{P}_{Y|X'}(y|x')$ compared to the true test labels Y' against the following standard discrimination measure derived from legal proceedings [26]:

$$J = \max_{d,d' \in \mathcal{D}} \left| P_{\hat{Y}|D}(1|d)/P_{\hat{Y}|D}(1|d') - 1 \right|. \tag{5.2}$$

For the equalized opportunities criterion, use the DEO as our metric. We compare the AUC and discrimination achieved by our algorithm with that of a method known as adversarial debiasing [133] (implementation given in [9]), which represents the current state of the art.

5.2.3 Experimental Results: Discrete Case

Figures 5-1 and 5-2 show the results of our experiments. For both datasets, in the case of independence, it can be seen that simply dropping the sensitive attribute D and applying logistic regression (LR) and random forest (RF) algorithms cannot ensure independence between $\hat{Y}$ and D. However, the proposed DTM-based algorithm provides a trade-off between performance and discrimination by varying the value of the regularizer λ in the optimization (4.19), which outperforms the optimized pre-processing methods in [19] on the Adult dataset, and achieves similar performance on the COMPAS dataset. More importantly, the DTM-based algorithm provides a smooth trade-off curve between the performance and discrimination, so that a desired level of fairness can be achieved by setting λ in practice. In addition, since our method only requires us to perform eigendecomposition, it runs significantly faster than the optimized pre-processing method, which needs to solve a much more complex optimization problem. Empirically, we find at least a tenfold speed up in runtime compared to the existing methods.

For separation, compared to naïve logistic regression, the proposed DTM-based algorithm dramatically decreases the DEO while maintaining similar AUC performance on both datasets, and outperforms the adversarial debiasing method in [133] on the Adult dataset. We note that the AUC and DEO curve achieved by the proposed algorithm in the separation setting has a smaller range compared to that in the independence setting. This is because the value of the regularizer λ is restricted in the separation optimization problem to $\lambda \in [0, 1)$, but only to $\lambda > 0$ for the independence optimization problem.

5.2.4 Experimental Setup: Continuous Case

In the continuous case, we experiment on the Communities and Crimes (C&C) dataset[3] [33, 102, 92]. The goal is to predict the crime rate Y of a community given a set of 121 statistics X (distributions of income, age, urban/rural, etc.). The 122-th statistic (percentage of black people in the community) is used as the sensitive variable D. All variables in this dataset are real-valued. The dataset was split into 1794 training and 200 test samples. Following Mary et al. [86], we use a neural net with a 50-node hidden layer (which we denote as $\Theta(x)$) and train a predictor $\hat{y} = T(\Theta(x))$ with the mean squared error (MSE) loss and the soft-HGR penalty, varying λ. For soft-HGR, we use two 2-layer NNs with scalar outputs as the two maximal correlation functions $\mathbf{g}$ and $\mathbf{h}$, and trained them according to (4.19) (independence) or (4.23) (separation). We then computed the test MSE and test discrimination metric in each case.

For demographic parity, our discrimination metric was $I(\hat{Y}; D)$, approximated using a standard mutual information estimator based on k nearest neighbors (kNN) [39]. For equalized opportunities, we computed $I(\hat{Y}; D|Y)$ using the same estimator.

We report the results of our experiment as well as that of the chi-squared method of Mary et al. [86] with the same architecture,[4] which we choose as the comparison method as it is one of the few methods designed to handle continuous D.

5.2.5 Experimental Results: Continuous Case

The results of the experiments are presented in Figure 5-3. As expected, we see a tradeoff between the MSE and discrimination, creating a frontier of possible values. We also see that the soft-HGR penalty provides modest gains compared to the chi-squared method for both independence and separation.

Moreover, our method runs significantly faster than the chi-squared method (on the order of seconds per iteration for our method versus just under a minute per iteration for the comparison method), as the chi-squared method requires computation over a mesh grid of a Gaussian kernel density estimation (KDE), which scales with the product of the number of "bins" (mesh points) and the number of training samples, while our method only scales with the number of samples ($O(n)$). For large bandwidths, the number of bins can become quite large. KDE methods also scale exponentially with dimensionality (see [121]), and thus if D is high-dimensional, the chi-squared method would run much slower than our method, which can take in an arbitrarily-sized input and scale linearly with the dimensionality of the input multiplied by the number of samples. Empirically, we find that our method runs around five times faster.

We also run experiments to illustrate how our method's simplicity allows it to adapt to the few-shot regime faster than that of the chi-squared method [60]. We take 10 few-shot samples from the training set, then train a network to predict Y from X without any fairness regularizer using the full training set. Then, we run 5 more iterations of gradient descent on the trained model using the fairness-regularized objective and the 10 few-shot samples, and compare the results between the soft-HGR and chi-squared regularizer. The results are shown in Figure 5-4.

Once again, we see the tradeoff curve, and see our method outperform the chi-squared method, and that it appears to be competitive with the standard case in just a few iterations, while the chi-squared method is still far from achieving the original

[3] `http://archive.ics.uci.edu/ml/datasets/communities+and+crime`

[4] Code for these experiments can be found at `https://github.com/jklee-mit/fair_independence_separation`.

MSE. We also significantly outperform the baseline (before fairness regularization) model in reducing discrimination, at the cost of only a small increase in error. Thus, in situations where, due to ethical/legal issues, only a few samples labeled with the sensitive attribute can be collected, fairness can still be enforced.

5.2.6 Discussion and Future Directions

The HGR maximal correlation provides us with a powerfully adaptive tool for handling data of different modalities under the two most common fairness criteria, performing competitively with state-of-the-art.

Moving forwards, it would be useful to expand the universality of this approach by trying to solve related fairness problems such as fair clustering [23], which also requires demographic parity, but has a primary clustering objective instead of classification or regression.

It is also important to remember that these measures of discrimination and performance have real-world consequences, and to ensure that these tools are used with those in mind. Determining the ideal tradeoff between fairness and accuracy requires a nuanced approach, which takes into account the full impact of both biases as well as poor performance on a system, and while one could argue that this is a problem for sociologists and legislators, an approach grounded in the statistics of the harm caused by biased systems should be considered when developing new policies and laws. Thus, it is of utmost importance that, for each application, one carefully considers the context in which these fairness algorithms are being employed, to ensure that they are used to benefit marginalized groups rather than cover up harm, as the next section illustrates.

5.3 Fair Selective Classification

While fair classification via demographic parity or equalized opportunities are common settings for fairness, fairness problems can arise in a number of different applications, and in some cases, can result in insidious situations where the fairness properties of a system can change depending on how its predictions are used.

One example sub-setting that exhibits such a phenomena is that of selective classification. Generally speaking, selective classification is a variant of the classification problem where a model is allowed to *abstain* from making a decision. This has applications in settings where making a mistake can be very costly, but abstentions are not (e.g., if the abstention results in deferring classification to a human actor).

In selective classification, a predictive system is given the choice of either making a prediction $\hat{Y}$ or abstaining from the decision. The core assumption underlying selective classification is that there are samples for which a system is more confident about its prediction, and by only making predictions when it is confident, the performance will be improved. To enable this, we must have a *confidence score* $\kappa(x)$ representing the model's certainty about its prediction on a given sample x [41]. Then, we threshold on this value to decide whether to make a decision or to abstain. We define the *coverage* as the fraction of samples for which we do not abstain on (i.e., the fraction of samples that we make predictions on).

As is to be expected, when the confidence is a good measure of the probability of making a correct prediction, then as we increase the minimum confidence threshold for making the prediction (thus decreasing the coverage), we should see the risk on the classified samples decrease or the accuracy over the classified samples increase. This leads us to the *accuracy-coverage* tradeoff, which is central to selective classification (though we note here the warning from the previous section about accuracy not telling the whole story).

Selective classifiers can work *a posteriori* by taking in an existing classifier and deriving an uncertainty measure from it for which to threshold on [41], or a selective classifier can be trained with an objective that is designed to enable selective classification [29, 127].

One common method of extracting a confidence score from an existing network is to take the softmax response $s(x)$ as a measure of confidence. In the case of binary classifcation, to better visualize the distribution of the scores, we define the confidence using a monotonic mapping of $s(x)$:

$$\kappa = \frac{1}{2} \log \left(\frac{s(x)}{1 - s(x)} \right) \tag{5.3}$$

which maps $[0.5, 1]$ to $[0, \infty]$ and provides much higher resolution on the values close to 1.

Finally, to measure the effectiveness of selective classification, we can plot the accuracy-coverage curve, and then compute the area under this curve to encapsulate the performance across different coverages [37].

However, Jones et al. [62] have shown that in some cases, when coverage is decreased, the difference in recall between groups can sometimes increase, magnifying disparities between groups and increasing unfairness. In particular, they have shown that in the case of the CelebA and CivilComments datasets, decreasing the coverage can also decrease the recall on the worst-case group. This, of course, has some very serious consequences for systems that require fairness, especially if it appears at first that predictions are fair enough under full coverage (i.e., when all samples are being classified).

This phenomenon can be mitigated by applying the sufficiency criterion to the learned features. The application of this criterion to fair selective classification comes to us by way of *calibration by group*. Calibration by group requires that there exists a score function $R = s(x)$ such that, for all $r \in (0, 1)$ [25],

$$P(Y = 1 | R = r, D = a) = r \ \ \forall a \in \mathcal{D}. \tag{5.4}$$

The following result from [8] links calibration and sufficiency:

Theorem 3. *If a classifier has sufficient features Θ, then there exists a mapping $h(\Theta) : \mathbb{R}^{d_\Theta} \to [0, 1]$ such that $h(\Theta)$ is calibrated by group.*

If we can find sufficient features $\Theta(X)$, so that the score function is calibrated by group based on these features, then we have the following result (the proof can be found in the Appendices):

Theorem 4. *If a classifier has a score function $R = s(x)$ that is calibrated by group, and selective classification is performed using confidence κ as defined in (5.3), then for all groups $d \in \mathcal{D}$ we have that both $A(\tau)$ and $\mathrm{PPV}(\tau)$ are* ***monotonically increasing*** *with respect to τ. Furthermore, we also have that $A(0) > 0.5$ and $\mathrm{PPV}(0) > 0.5$, where $A(\tau)$ is the selective accuracy at threshold τ and $\mathrm{PPV}(\tau)$ is the selective precision $\mathbb{P}(Y = 1|\hat{Y} = 1)$ at threshold τ.*

From this, we can guarantee that as we sweep through the threshold, we will never penalize performance of any one group in service of increasing the overall precision. Furthermore, in most real-world applications, the precision on the best-performing groups tends to saturate very quickly to values close to 1 when coverage is reduced, and thus, if we can guarantee that the precision increases on the worst performing group as well, then in general, the difference in precision between groups decreases as coverage decreases.

For a more detailed analysis of fair selective classification using sufficiency, please see Appendix B.

5.3.1 Experimental Setup

To evaluate our method, we look to see how the tradeoff between coverage and precision of a selective classifier trained using our regularizer compares to that of other methods. We test our method on three datasets commonly used in fairness: Adult, CelebA[5], and CivilComments[6]. In all cases, we use the standard train/val/test splits packaged with the datasets and implement our code in PyTorch. We set $\lambda = 0.7$ for all datasets as well, which we chose by sweeping over values of λ across all datasets. The code for this method can be found in the UQ360 toolbox [44].

For the Adult dataset [65], we use the full dataset. The data X consists of demographic information about individuals, including age, education, marital status, and country of origin. Following [9], we one-hot encode categorical variables and desig-

[5] http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

[6] https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification/data

Table 5.1: Summary of datasets.

Dataset	**Modality**	**Target**	**Attribute**
Adult	Demographics	Income	Sex
CelebA	Photo	Hair Colour	Gender
CivilComments	Text	Toxicity	Christianity

nate the binary-quantized income to be the target label Y and sex to be the sensitive attribute D. In order to induce a bias phenomenon in the dataset, we also drop all but the first 50 samples for which $D = 0$ and $Y = 1$. We then use a 2-layer neural network with 80 nodes in the hidden layer for classification, as in [86], with the first layer serving as the feature extractor and the second as the classifier, and train the network for 20 epochs.

The CelebA dataset [80] consists of 202,599 images of 10,177 celebrities, along with a list of attributes associated with them. As in [62], we use the images as our data X (resized to 224x224), the hair color (blond or not) as the target label Y, and the gender as the sensitive attribute D, then train a ResNet-50 model [54] (with initialization using pre-trained ImageNet weights) for 10 epochs on the dataset, with the penultimate layer as the feature extractor and the final layer as the classifier.

The CivilComments dataset [15] is a text-based dataset consisting of a collection of online comments, numbering 1,999,514 in total, on various news articles, along with metadata about the commenter and a label indicating whether the comment displays toxicity or not. As in [62], we let X be the text of the comment, Y be the toxicity binary label, and D to be mention of Christianity. We preprocess the data by passing them through a BERT model [31] with Google's pre-trained parameters [114], and treat the output features as the pre-processed input into our system. We then apply a 2-layer neural network to the pre-processed inputs with 80 nodes in the hidden layer, once again treating the layers as feature extractor and classifier, respectively. We train the model for 20 epochs.

We compare our results to a baseline where we only optimize the cross-entropy loss, as in standard classification. We also compare our method to the group distributionally robust optimization (DRO) method of [106, 112], using the code provided publicly on Github[7], which has been shown to mitigate the disparity in recall rates between groups in selective classification [62], as well as a mutual information upper-bound regularizer outlined in Section B.3.

5.3.2 Experimental Results

Figure 5-6 shows the group-specific precisions for the Adult dataset. We observe that, for the baseline method, the increase in overall performance comes at the cost of worse performance on the worst-case group. This phenomenon is heavily mitigated in the case of DRO, but there is still a gap in performance in the mid-coverage regime. Our HGR method also shows a gap in performance, compared to the mutual information-based method, which shows the precisions converging to equality very quickly as coverage decreases. We observe a similar hierarchy of performances in the other datasets (Figures 5-7 and 5-8).

In order to numerically evaluate the relative performances of the algorithms for all the datasets, we compute the following quantities: area under the average accuracy-coverage curve [37] and area under the absolute difference in precision-coverage curve (or area between the precision-coverage curve for the two groups). Table 5.2 shows the results for each method and dataset.

From this, it is clear that the performance of the HGR method sits in between that of the DRO method and the mutual information-based method, reflecting the fact that the HGR specifically optimizes for sufficiency (while the DRO method optimizes for equalized opportunities), but the mutual information-based method uses a tighter approximation of the quantities we desire.

[7] `https://github.com/kohpangwei/group_DRO`

Table 5.2: Area under curve results for all datasets.

Dataset	Method	Area under accuracy curve	Area between precision curves
Adult	Baseline	0.931	0.220
	DRO	0.911	0.116
	MI	0.887	0.021
	HGR	0.91	0.187
CelebA	Baseline	0.852	0.094
	DRO	0.965	0.018
	MI	0.975	0.013
	HGR	0.932	0.014
CivilComments	Baseline	0.888	0.026
	DRO	0.944	0.013
	MI	0.943	0.010
	HGR	0.937	0.014

5.3.3 Discussion and Future Directions

While the HGR Maximal Correlation can sometimes perform competitively with state-of-the-art methods, we also see that it can sometimes be outperformed by other methods. This application illustrates the weakness in our formulation for enforcing sufficiency, and shows that a proper bound on the mutual information can be more effective at times, though in some cases their performances are comparable.

However, beyond the performance, the very existence of this problem of fair selective classification reveals that machine learning systems can have very dangerous blind spots when deployed in the real world. A minor modification to an existing system, such as thresholding decisions on a confidence value, can result in biases and disparities being magnified, which serves as a warning that each new application must be closely scrutinized, no matter how much it appears to be similar to existing problems. Failure to do so could result in vital systems failing the most vulnerable people in our society, which we are sworn to protect, and even worse, can lead to a false sense of complacency if the wrong fairness measure is applied.

Continual vigilance by humans is, at the moment, the only way to avert these issues. Perhaps in the future a more universal measure of fairness might be developed, which can close these gaps in biases in different settings, but for now, all system designers have an ethical responsibility to think carefully through the ramifications of their creations.

5.4 Privacy

As a second application for sensitivity-aware learning, one important consideration when sending sensitive information is that of privacy. There are many legal and ethical requirements for the transfer of personal data that requires certain aspects of the person remain anonymous, and numerous examples exist of cases where data that was meant to be anonymized was found to contain sensitive information about the individuals from which the data was collected [93].

A number of different definitions for privacy have arisen in recent years [113, 108, 42, 34], but we will focus on one that is especially suited for our method: the privacy funnel.

The original form of the privacy funnel problem is as follows [83]:

$$\begin{aligned} \min_{\Theta(X)} \quad & I(D;U) \\ \text{s.t.} \quad & I(X;U) \geq R, \end{aligned} \tag{5.5}$$

where U are the features learned by Θ.

This can be interpreted as finding features of the data X that preserve at least some minimum amount of information about the data while minimizing the information that these features "leak" about some variable D. For example, X could be financial data about an individual, and D could be identifying information such as age, race, gender, etc.

A more directed variant of this problem comes to us through [18]. In this case, we assume that the features we learn from X are to be used to predict some label Y, while still trying to minimize the information that leaks about D. Then, the learning problem is:

$$\begin{aligned} \min_{\Theta(X)} \quad & I(D;U) \\ \text{s.t.} \quad & I(Y;U) \geq R. \end{aligned} \tag{5.6}$$

We draw connections between this setup and that of learning under the independence constraint. Specifically, by adjusting the value of λ on the relaxed independence learning problem

$$\max_{\substack{\Theta\colon \mathcal{X}\to\mathbb{R}^m \\ T\colon \mathbb{R}^m\to\mathcal{Y}}} \mathrm{HGR}(U;Y) - \lambda \mathrm{HGR}(U;D), \tag{5.7}$$

we can produce a frontier of possible values for $I(D;U)$ vs. $I(Y;U)$, from which we can find the optimal value of $I(D;U)$ for a given constraint on $I(Y;U)$.

5.4.1 Experimental Setup

To examine the privacy-utility tradeoff of our independence-based HGR method, we test on the Adult and COMPAS datasets.

For the Adult dataset [65], we once again use the full dataset, with one-hot encoding of categorical variables. Again, we designate the binary-quantized income to be the target label Y and sex to be the sensitive attribute D. For the COMPAS dataset [100] we also use the full dataset as given in the AIF360 fairness toolkit [9]. We once again set the target Y to be recidivism, while the sensitive attribute D is race (we filter for only Caucasian and African-American individuals). Our data X consists of the remaining variables.

In both cases, we use a two-layers neural network with 80 nodes in the hidden layer for classification, as in [86], with the first layer serving as the feature extractor and the second as the classifier, and trained the network for 20 epochs. We apply the independence regularizer to the feature layer to enforce independence between the features and the sensitive attributes, then compute the mutual information quantities $I(Y;U)$ and $I(D;U)$.

5.4.2 Experimental Results

The results of our experiments can be found in Figure 5-9. This time, we can see a clear gain in privacy for our HGR-based method as compared to the chi-squared regularizer.

We also see the frontier of possible values once again. The similarities between this setup and the fairness setup can be seen quite clearly, with our method working effectively to solve both problems. In this case, we also see a significant (approximately five-fold) decrease in runtime for our method compared to the chi-squared method.

5.4.3 Discussion and Future Directions

Once again, we see how the independence criterion, along with our HGR formulation for enforcing it, can be applied to solve an important real-world problem. In this, we can see that these criteria have multiple uses. Thus, adapting the HGR to be able to enforce them not only allows us to solve one important problem, but a whole host of them.

However, the privacy funnel is not the only formulation of privacy. Moving forwards, it would be remiss not to consider the concept of differential privacy, which has become very popular of late [32, 113, 108, 42, 34]. Differential privacy has connections with the privacy funnel [82, 107], but a direct equivalence between the two has not yet been shown. Being able to draw a stronger connection between these two quantities could be the key to allowing us to use the HGR to enforce differential privacy as well.

5.5 Domain Generalization

The sufficiency criterion also has a useful application in the field of transfer learning, particularly that of domain generalization.

Consider the case where we have n domains, which we denote with $D = 1, ..., n$.

We also have training samples (x_i, y_i, d_i) from these domains labeled with the domain they were extracted from.

We wish to learn a feature mapping $\Theta(X)$ that will do well on some new domain $D = n + 1$, and we assume that a mapping that performs well on all n training domains will do well on the $(n + 1)$th domain.

For this, we would like to learn a mapping that performs well across all domains, and which captures the aspect of each class that is independent of the domain, that is, we wish to learn features that are domain *invariant*.

This is the principle behind *invariant risk minimization (IRM)*, which seeks to learn a $\Theta(X)$ such that $P(Y|\Theta(X), D) = P(Y|\Theta(X))$ [88, 5], corresponding to the sufficiency condition $Y \perp D|U$, where U are the features produced by Θ on X.

5.5.1 Experimental Setup

We test our sufficiency-based regularizer on the RotatedMNIST and ColoredMNIST datasets, as found in the DomainBed benchmark suite[8] [48].

The MNIST dataset[9] [71] consists of 60,000 training and 10,000 testing images of handwritten digits, from 0 to 9. The images are grayscale, and there are between 5,000 and 7,000 training images for each class, as well as between 890 and 1,140 testing images for each class [52]. The RotatedMNIST and ColoredMNIST datasets are synthetic datasets built from MNIST.

The RotatedMNIST dataset [43] consists of six domains, each corresponding to a different degree of roll rotation of the original MNIST images: 0°, 15°, 30°, 45°, 60°, and 75°. Each domain contains the same subset of 1,000 images sampled across all 10 labels, rotated accordingly. The purpose of this dataset is to learn rotation-invariant representations of the digits. During training, five of these domains are used for training, with the sixth domain used for testing.

The ColoredMNIST dataset [5] is a binary classification dataset that is generated as follows: assign an initial label y' to each image based on the digit: 0 for digits

[8] `https://github.com/facebookresearch/DomainBed`

[9] `http://yann.lecun.com/exdb/mnist/`

Table 5.3: Domain generalization accuracies on the RotatedMNIST dataset for each test domain. Values are reported with 95% confidence intervals over 5 trials.

Method	**0°**	**15°**	**30°**	**45°**	**60°**	**75°**
ERM	93.8 ± 0.4	98.5 ± 0.3	98.6 ± 0.2	98.8 ± 0.3	98.2 ± 0.2	94.8 ± 0.5
IRM	94.0 ± 0.2	91.7 ± 0.9	93.9 ± 1.6	93.7 ± 0.5	92.8 ± 0.2	91.3 ± 2.3
MI	92.6 ± 0.5	98.0 ± 0.2	98.1 ± 0.3	97.7 ± 0.1	97.7 ± 0.0	92.8 ± 0.9
HGR	93.2 ± 0.2	98.2 ± 0.0	98.6 ± 0.3	98.3 ± 0.5	98.4 ± 0.1	93.9 ± 0.4

0-4, and 1 for digits 5-9. Then, the final label y is produced by randomly flipping y' with probability 0.25. The environment e is obtained by flipping y with probability p_d, which is defined by the domain d. The color of the image is then set to red if $e = 1$ and green if $e = 0$. The domains used in testing have flip probabilities $p_0 = 0.1, p_1 = 0.2, p_2 = 0.9$. Thus, in the domain $d = 2$, the colors are mostly flipped compared to the other two domains, and an invariant predictor should use the shape of the digit rather than the colour for predicting y. During training, two of these domains are used for training, with the third domain used for testing.

For each dataset, we use a 4-layer convolutional neural net as our feature extractor, and a fully connected 2-layer network as our final classifier, and train for 2000 random batch iterations. We compare our method with IRM, as well as the mutual information upper-bound found in Section B.3, and a baseline in which we train using empirical risk minimization (ERM) and ignore domain labels.

5.5.2 Experimental Results

Table 5.3 summarizes our results on the RotatedMNIST dataset. We can see that all of the algorithms perform approximately on par with one another, with no one method dominating over the majority of domains.

Table 5.4 summarizes our results on the standard ColoredMNIST dataset. We can see that in the case where the training domains are $p = 0.1$ and $p = 0.2$, with test domain $p = 0.9$, IRM fails to generalize and learn the shape of the digits, relying on the colors for classification. Our HGR method and the mutual information regularizer

Table 5.4: Domain generalization accuracies on the three-domain ColoredMNIST dataset for each test domain. Values are reported with 95% confidence intervals over 5 trials.

Method	$p = 0.1$	$p = 0.2$	$p = 0.9$
ERM	70.3 ± 0.3	73.0 ± 0.1	19.0 ± 5.4
IRM	69.6 ± 0.4	72.6 ± 0.5	10.3 ± 0.1
MI	68.7 ± 0.7	60.8 ± 7.7	49.2 ± 0.2
HGR	71.0 ± 0.6	71.7 ± 0.3	50.3 ± 0.5

Table 5.5: Domain generalization accuracies on the four-domain ColoredMNIST dataset for each test domain. Values are reported with 95% confidence intervals over 5 trials.

Method	$p = 0.1$	$p = 0.2$	$p = 0.8$	$p = 0.9$
ERM	69.8 ± 1.4	69.8 ± 0.1	69.7 ± 0.3	68.2 ± 0.1
IRM	67.8 ± 1.5	69.8 ± 1.3	70.0 ± 0.9	69.8 ± 0.7
MI	69.4 ± 0.8	70.7 ± 0.2	71.4 ± 0.1	66.9 ± 0.4
HGR	68.7 ± 0.6	70.2 ± 0.7	69.7 ± 0.4	67.5 ± 0.6

cannot quite force the system to learn to identify the shapes, but it does recognize when the color is an environmental factor and thus avoids using it for classification, resulting in around 50% accuracy. However, our method does perform better on the other test domains, suggesting that it is less punitive on the overall performance compared to the mutual information regularizer.

If we add an extra domain that is similar to $p = 0.9$ (namely, $p = 0.8$), we see in Table 5.5 that all the methods now generalize much better and can perform with better than 50% accuracy on all domains, suggesting that they are learning shape-based features as intended. However, the ERM method is also able to do this, which would imply that domain generalization techniques are not necessary in this case.

5.5.3 Discussion and Future Directions

In this section, we have shown that the HGR maximal correlation performs competitively with other state-of-the-art methods for sufficiency regularization. This suggests that there are applications in which the approximation used in the regularizer is close enough to be effective, likely due to the fact that Y and D are much more cleanly separated, with a clear definition of what an invariant feature would be in this case compared to the fairness application.

Moving forwards, testing on more complex domain generalization datasets such as VLCS [35], PACS [73], DomainNet [97], and WILDS [64] may help illuminate more connections between the ability of different methods to capture different types of invariances needed to generalize to different domains. In addition, there is also the debate as to whether or not invariance is the correct paradigm for domain generalization [104], and so some investigation into what properties enable this approach to work would also be warranted.

Finally, we note that this application also illustrates the potential power in these regularizers, with their ability to traverse highly disparate fields and find connections between them.

5.6 Concluding Remarks

In this chapter, different problems from different areas of machine learning have been shown to break down to the same objectives, suggesting that while the growth of new machine learning applications continues, some of this flood of new problems will have strong enough ties to older problems that a framework such as the HGR can be used to rapidly adapt to them with little difficulty.

In some cases, the HGR is well-suited to solving these problems. In others, it is competitive, but does not quite perform as well as state-of-the-art. There is an argument to be made that this slight gap in performance is acceptable given how quickly the HGR can be adapted to a new problem. After all, a solution in the present is much more valuable than a potentially better solution in the future.

Moving forward, one might speculate that the diversity of problems solved by these criteria may indicate that developing some underlying framework to view all machine learning problems and categorize them across disparate fields may yield a useful taxonomy that can help those in the future to more quickly find the space of possible solutions in which to explore. While some domain expertise will always be necessary, planting the seeds of looking into such problems with a holistic perspective may allow for new avenues of cooperation and the ability to leverage new resources that might not have seemed relevant to the untrained eye.

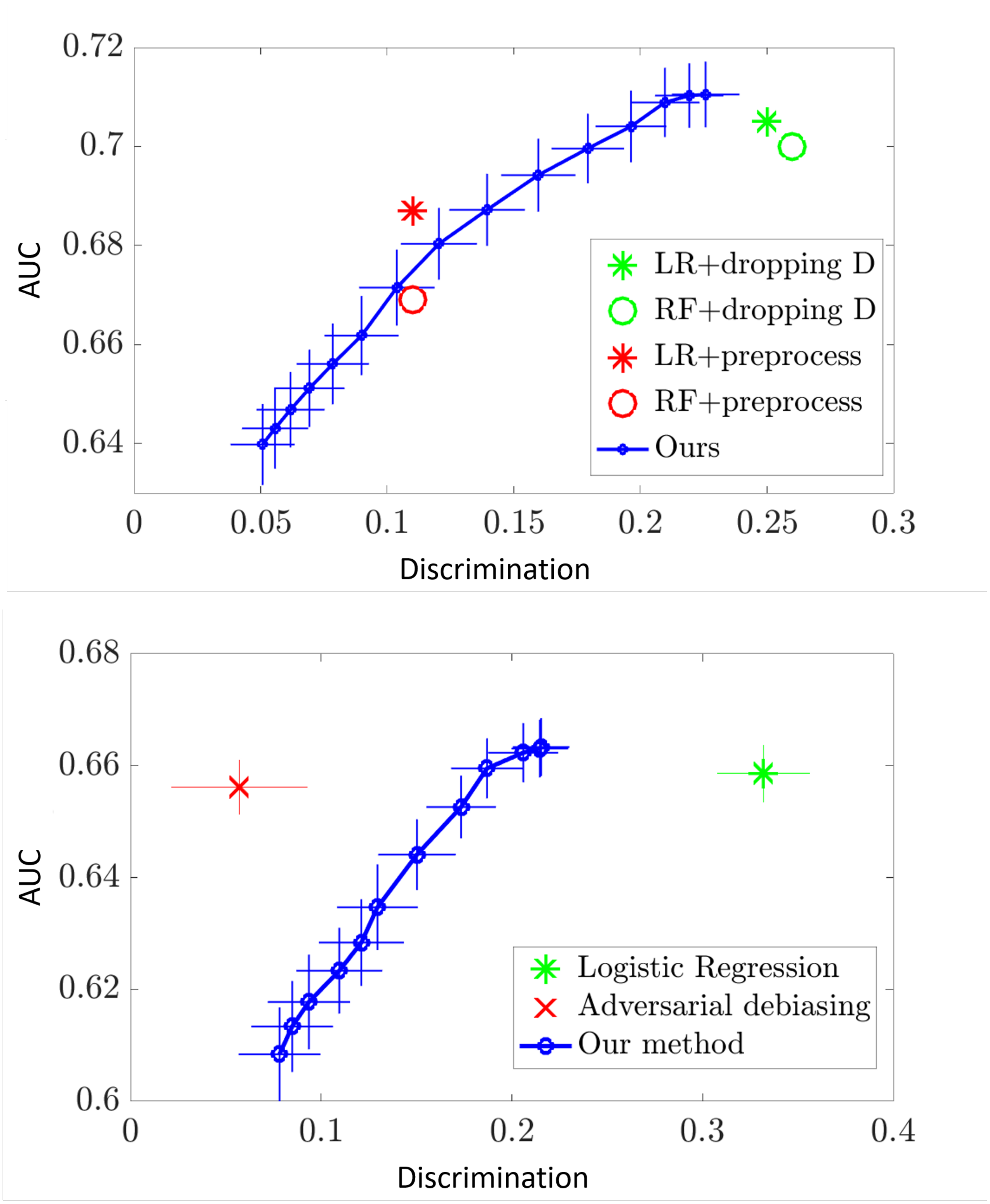


Figure 5-1: Regularization results on COMPAS dataset, with AUC plotted against the discrimination measure for independence (top) and separation (bottom), respectively.

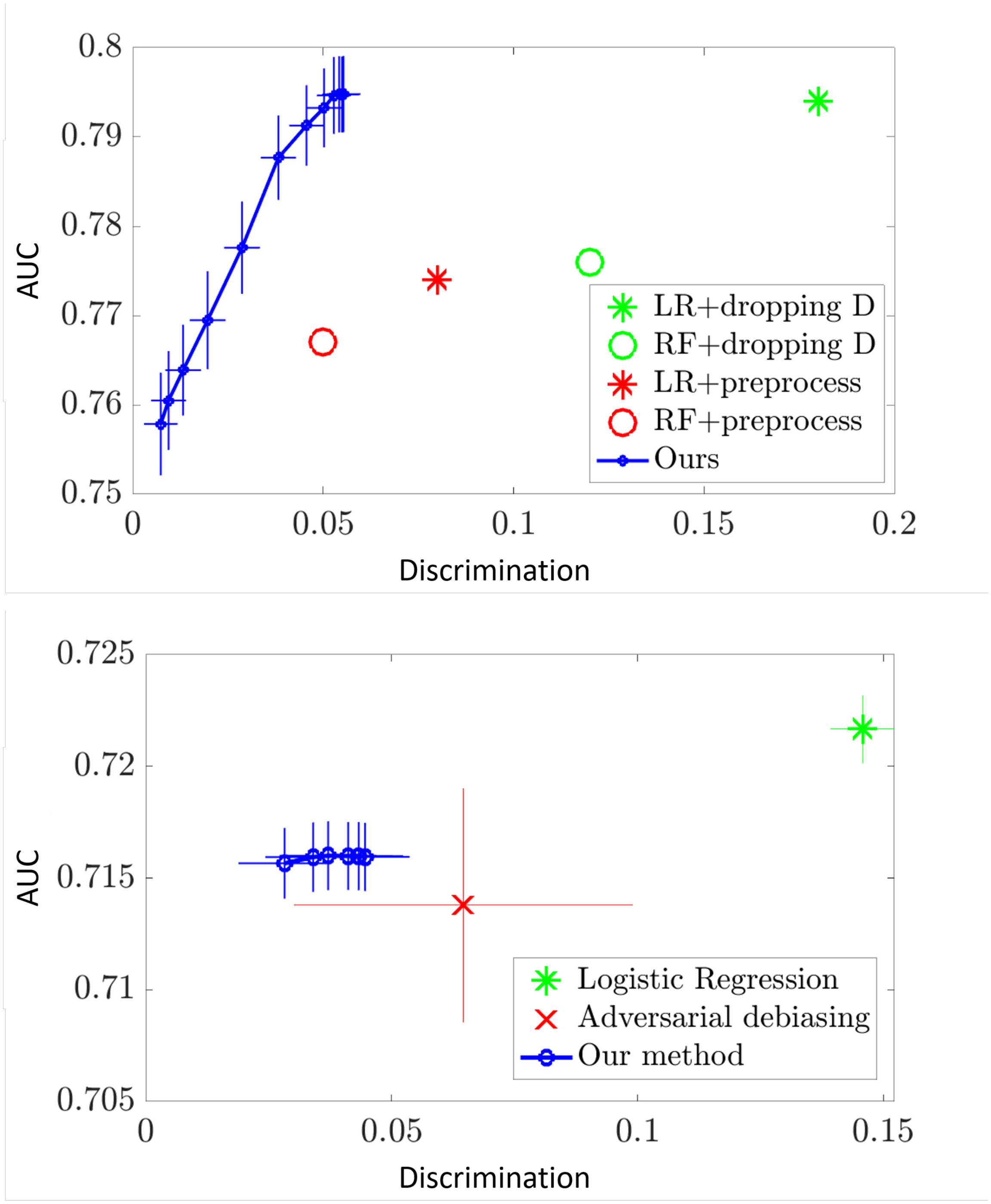


Figure 5-2: Regularization results on Adult dataset, with AUC plotted against the discrimination measure for independence (top) and separation (bottom), respectively.

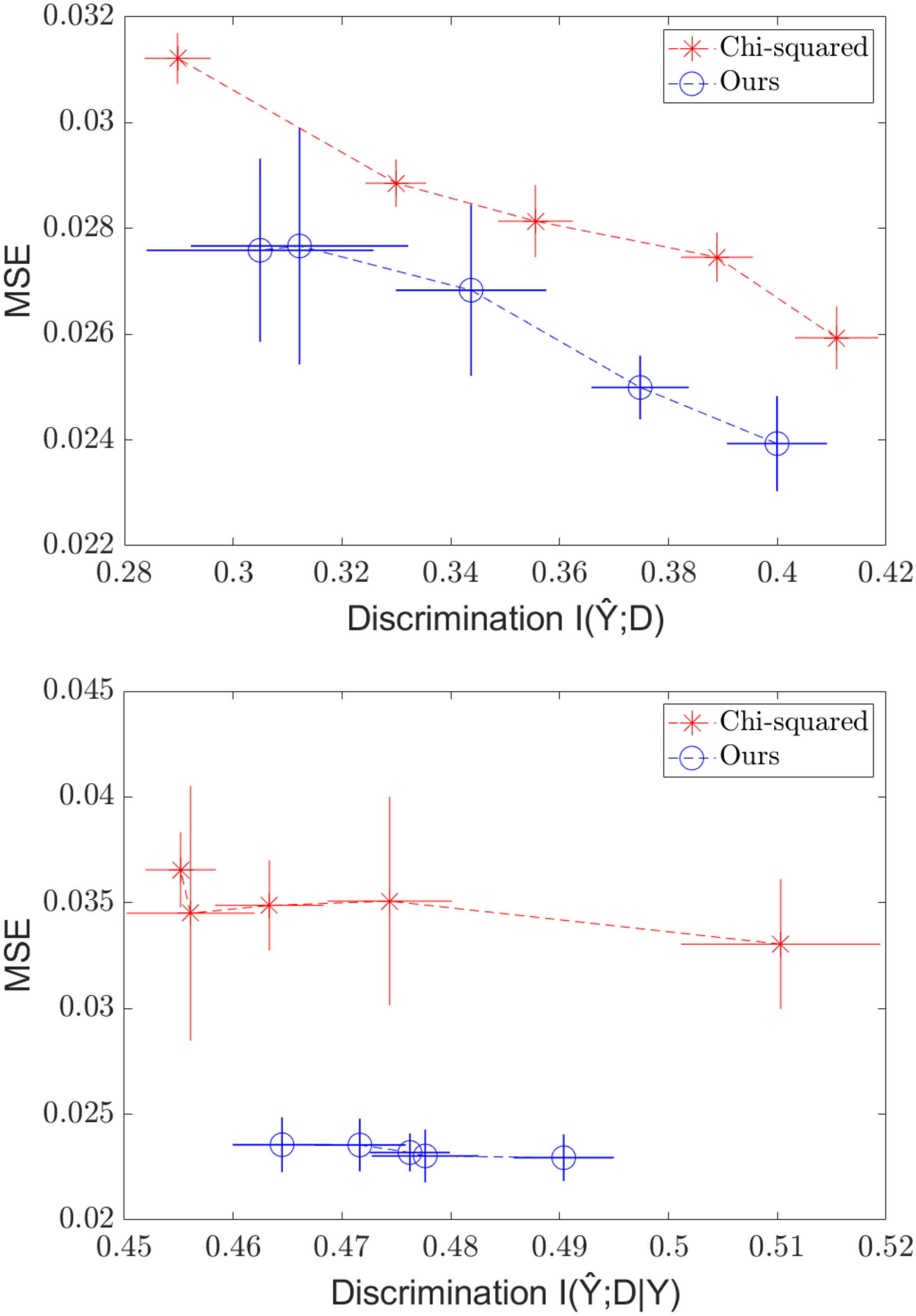


Figure 5-3: independence (top) and separation (bottom) regularization on the C&C dataset, with MSE plotted against $I(\hat{Y}; D|Y)$.

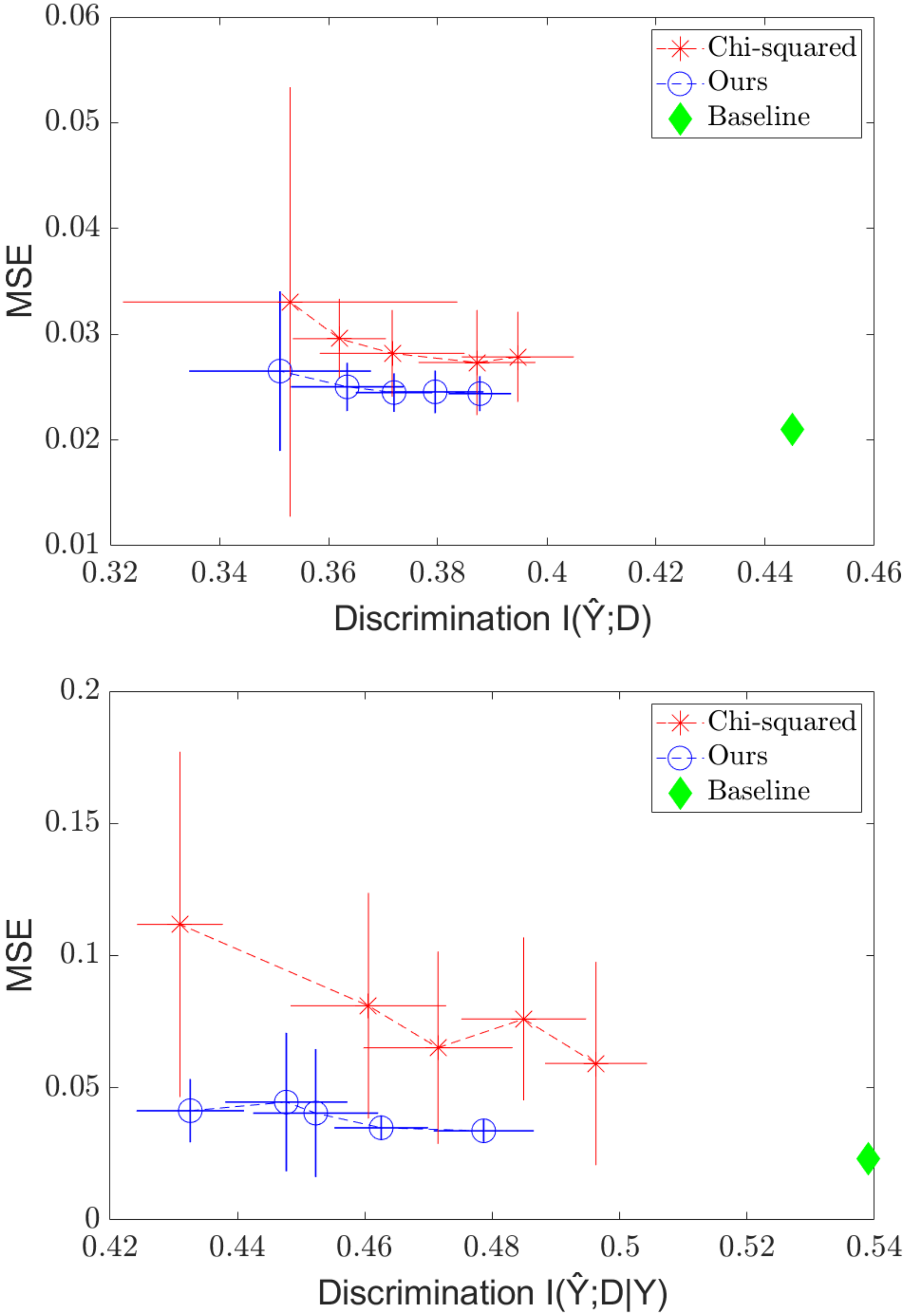


Figure 5-4: Few-shot independence (top) and separation (bottom) regularization on the C&C dataset in the settings, with MSE plotted against $I(\hat{Y}; D|Y)$.

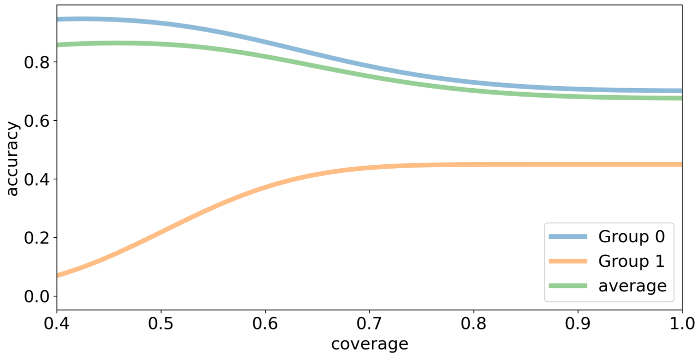
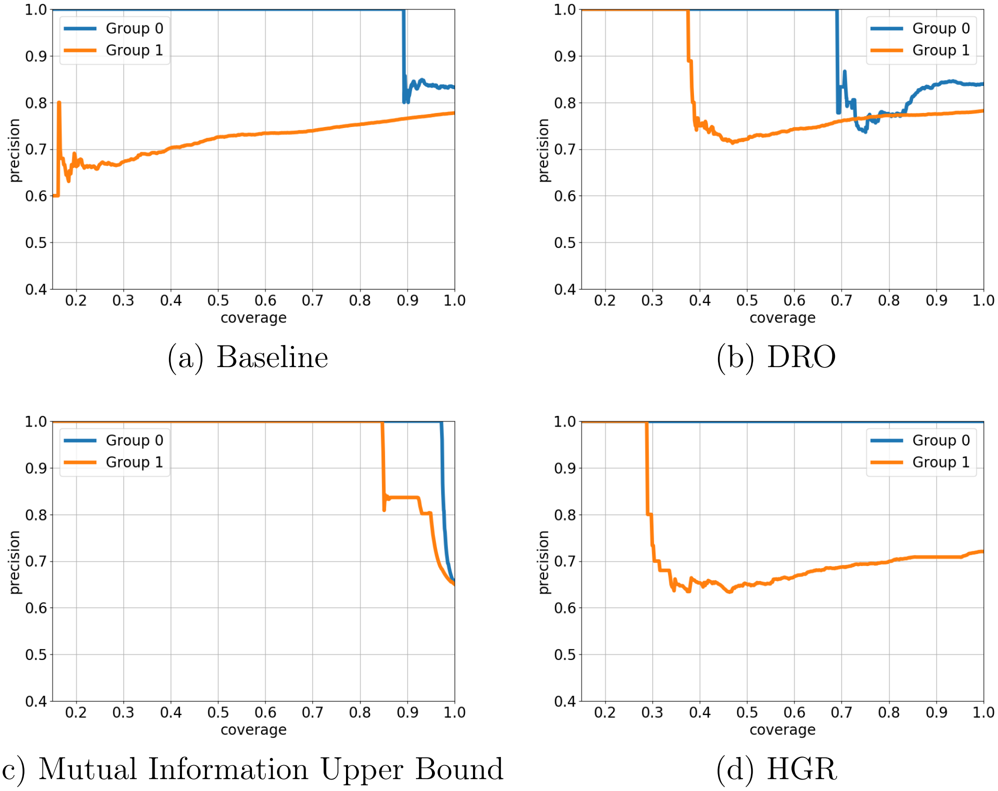


Figure 5-5: While selective classification can improve overall accuracy as coverage decreases, on certain protected groups the performance can decrease instead.



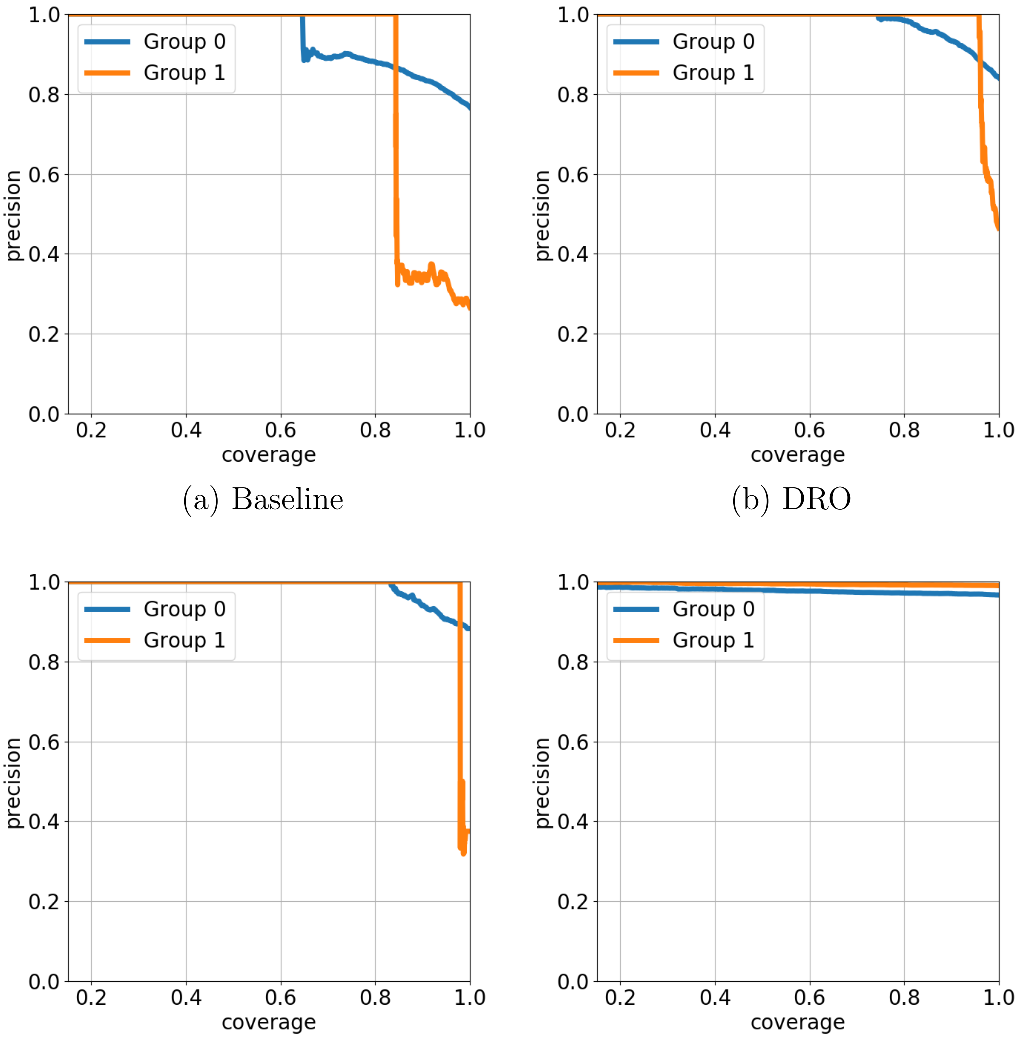
(a) Baseline

(b) DRO

(c) Mutual Information Upper Bound

(d) HGR

Figure 5-6: Group-specific precision-coverage curves for Adult dataset for the four methods.



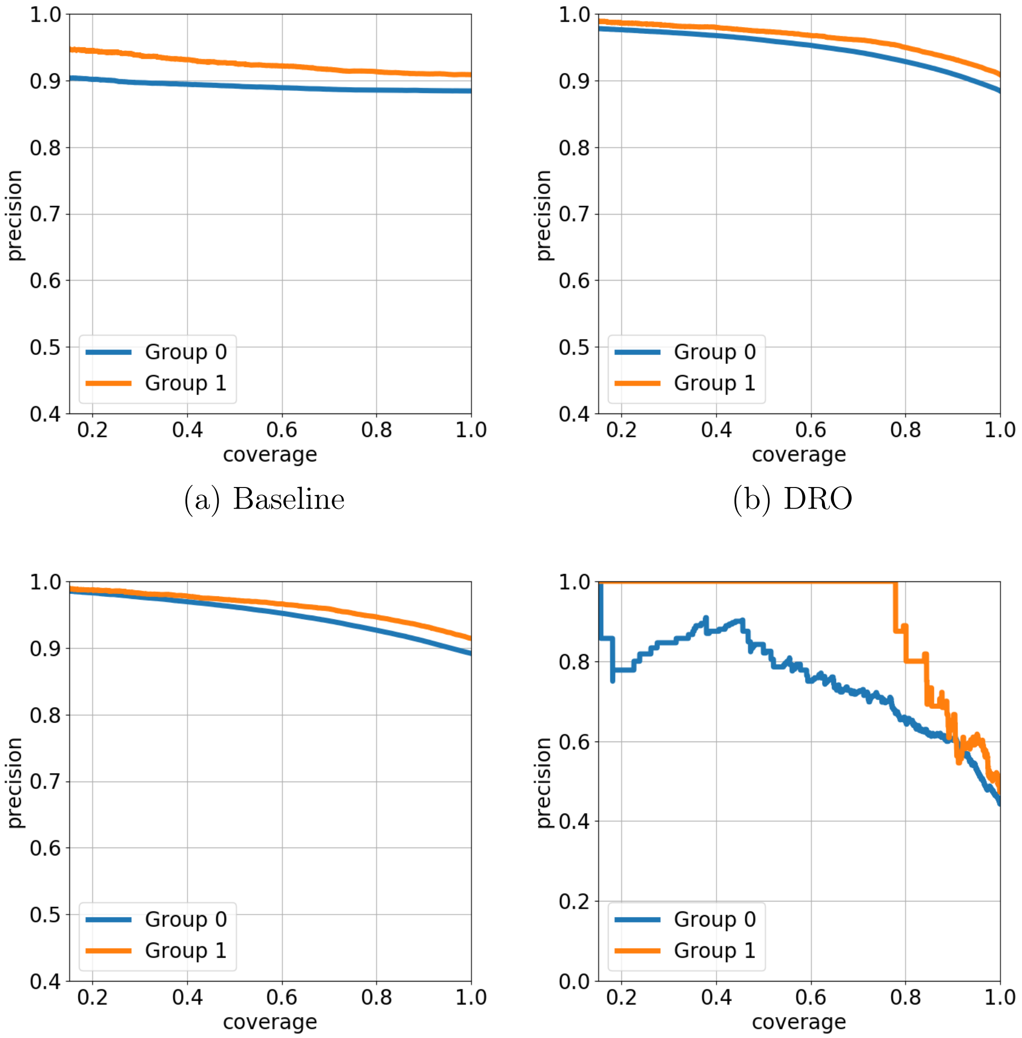
(a) Baseline

(b) DRO

(c) Mutual Information Upper Bound

(d) HGR

Figure 5-7: Group-specific precision-coverage curves for CelebA dataset for the four methods.



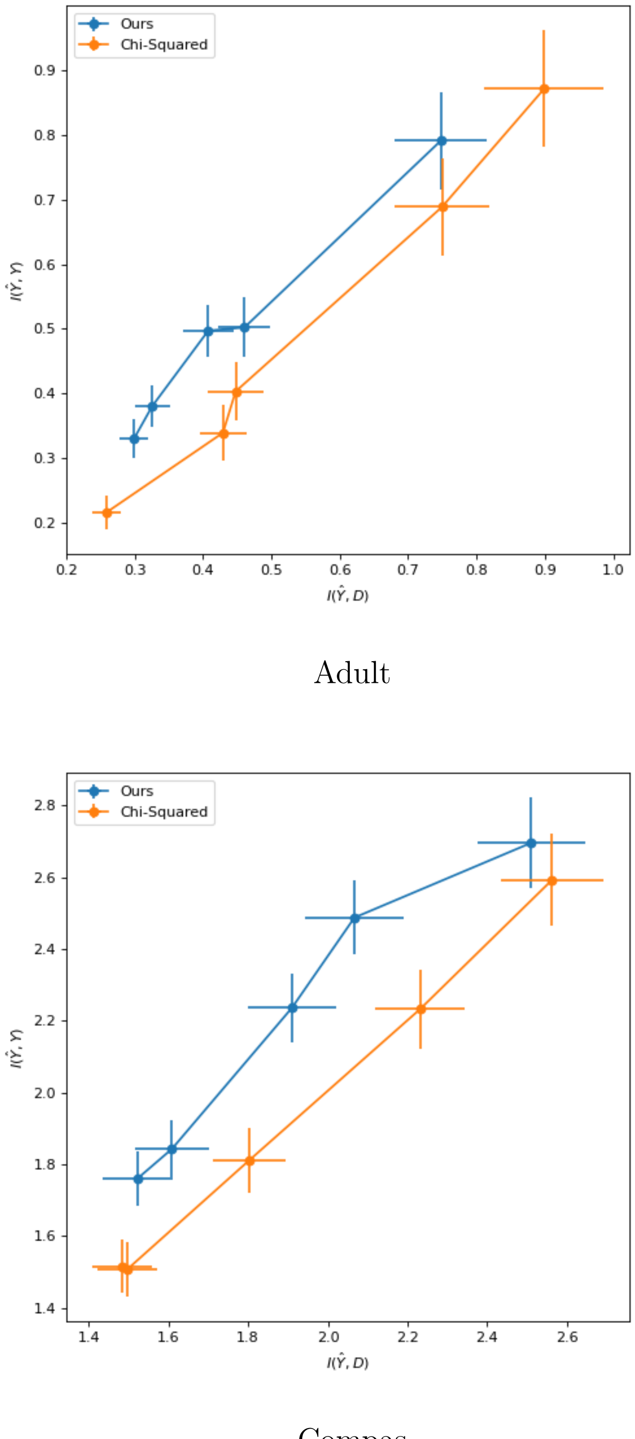
(a) Baseline

(b) DRO

(c) Mutual Information Upper Bound

(d) HGR

Figure 5-8: Group-specific precision-coverage curves for CivilComments dataset for the four methods.



Adult

Compas

Figure 5-9: Privacy-utility tradeoff curves on the Adult and COMPAS datasets for our HGR-based method compared to the Chi-squared regularizer method. Values are reported with 95% confidence intervals over 5 trials.

Chapter 6

Outlier Detection

Thus far, we have explored the use of the HGR in solving problems involving feature selection for transfer learning and feature suppression via sensitivity criteria. However, the HGR can also be used to solve problems in which we wish to extract features to perform secondary tasks, in addition to suppressing features that interfere with a secondary objective or simply selecting a subset of features for another task.

One such example where we desire to extract features with a secondary task in mind is found in the problem of outlier detection. In some ways, we can view this problem as the opposite of that of domain generalization; rather than trying to perform well on new kinds of data by assuming them to be somehow similar to existing data, we instead label these novel samples as "outliers."

One can imagine that, depending on the application, one approach may be preferred over another. While a classifier designed to classify types of cars should be able to operate in different weather "domains", any sort of medical device should immediately raise a red flag if it detects something outside its normal mode of operation.

However, even if we wish to detect outliers, there remains the open problem in how we define an outlier. We shall see in this section that the HGR can provide some insights into this question, by taking a self-supervised perspective on the problem.

6.1 Problem Definition

Very broadly, we can think of the problems of anomaly, outlier, and out-of-distribution (OOD) detection as follows: given an *in-distribution* $P(x)$ (or samples from which to learn the in-distribution) and a candidate data point, determine if the data point was generated by $P(x)$ or if it is an "outlier."

Immediately, we see that the definition of an outlier is heavily dependent on the choice of $P(x)$. This, by necessity, must depend on the application in question, as we can imagine a number of different scenarios involving the same data in which different groups could be considered the outliers.

As a simple example, consider the case where our input data (both inlier and outlier) consists of both the SVHN and MNIST datasets in their entirety. The Street View House Numbers (SVHN) dataset [89] consists of labeled images of the digits 0 to 9 in the real world, obtained from house number images in Google Street View. Meanwhile, the MNIST dataset, as stated previously, is a dataset of handwritten digits from 0 to 9 as well.

Given these data, with no other additional information, it is impossible to determine what is and isn't an outlier. One could imagine that the digits 1 to 9 are inliers, and the digit 0 is an outlier. Or we could assume that the handwritten digits are the inlier set and photos are the outlier set. Or maybe all of the data are inliers and the outliers are any images that are not of digits. All of these are valid interpretations of the problem.

Traditionally, outliers are categorized into three main "flavours":

- Samples drawn from a different domain, usually known as out-of-distribution samples. For example, for a classification problem involving the classification of photos, this could include paintings or sketches of the classes. [55, 79]

- Samples drawn from the same domain as the in-distribution, but from a different class. For example, photographs of fish when the task is to classify mammals. [68, 55, 79]

- Samples that represent anomalous behaviour of what the in-distribution models. For example, an image of a defective product on a manufacturing line, where the in-distribution is the set of images of non-defective products. [68, 13]

These three types of outliers are illustrated in Figure 6-1. Once again, we could imagine applications in which some or all of these would not be considered outliers.

In the first case, if our goal is simply to identify "planes", then perhaps a drawing of a plane would be a reasonable input. For the second, a vehicle classifier would not consider a car to be an outlier, even if it had only seen planes so far. And as for the third image, if the expectation is that some images are from a manufacturing plant, then a part of a plane could also be something to expect.

In any practical application, which set of outliers we wish to detect will be defined according to the type of consequences one is trying to avoid. For example, if we are trying to perform automatic screening for defective parts in an assembly line or disease in humans, then the most dangerous outliers would represent highly irregular versions of the samples in the inlier dataset that could confuse a classifier. Meanwhile, a change in domain might be useful for a system that is designed to only operate in one environment, in order to detect a change in operating conditions (e.g., a vehicle classifier that accidentally shifted and is now recording pedestrians instead).

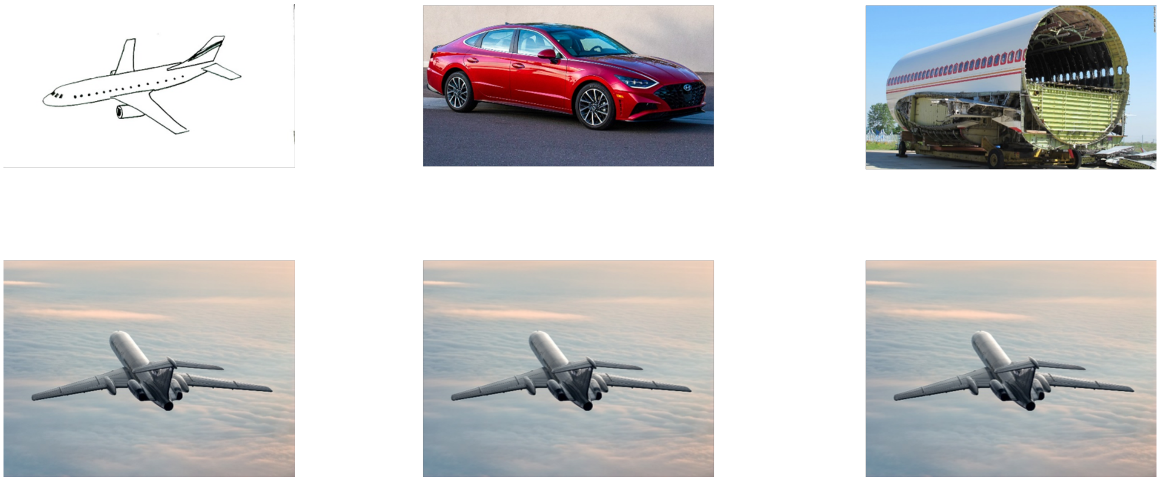


Figure 6-1: Different types of outliers may require different methods of detection.

Thus, in order to be adaptable to different situations, we must have some way of specifying what defines an outlier, or alternatively, what defines $P(x)$. The latter can be done through samples that are inliers, but if the samples do not cover the entire space, then parts of the inlier space may be classified as anomalies. An alternative approach is to provide samples that are representative of the outliers we would expect to encounter, and ensure that these samples are classified as outliers [55]. This method shares a similar risk of not properly covering the space of outliers, especially since we often have few examples of outliers, and no examples of certain types of outliers, but it can help guide anomaly detectors in learning the common types of outliers to expect and perform better on identifying them.

In the first case of outliers detailed above, we may also have out-of-distribution samples drawn from some other domain besides the in-distribution domain, and the objective will be to learn a detector that generalizes to other domains in testing (e.g., if the in-distribution is the set of photographs, then one might be given a large collection of sketches as out-of-distribution samples, with the hopes of generalizing to detecting paintings as out-of-distribution as well).

In particular, we also assume that outlier detection is an auxiliary objective, and that we have a primary objective whereby we seek to predict some target Y from X, while identifying outlier samples that are not drawn from $P(X)$.

6.1.1 Prior Work

Anomaly detection has a long history, with deep learning methods only recently being used to solve this problem in fields such as computer vision. Most of these methods have two components, which may or may not be implicitly integrated into one. The first is an "anomaly score," which can be computed from a sample, usually using some features or predictions from a neural network. The second is the network itself, which may need to be specially chosen to produce the scores, or simply be a modified version of an existing architecture with an additional regularizer that highlights the anomalies.

Kwon et al. [68] and Bergmann et al. [13] use auto-encoders to learn latent representations of data, then derive an anomaly score related to the reconstruction error, with the assumption that outliers, which are more poorly modeled by the autoencoder, will thus be more difficult to reconstruct.

Hendrycks et al. [55] and Liu et al. [79] both assume a primary classification task for which outlier detection is desired as a secondary objective. Both methods modify the training of a neural net for the classification task with a regularizer that pushes inlier samples to have more extreme values in the prediction layer, and for outlier samples to be spread out thinly across the final activation layer. Winkens et al. [123] does away with the need for OOD samples at training time, by using a contrastive loss to force the network to learn a better model for the in-distribution.

6.2 Computing Features for Outlier Isolation

In order to compute features which enable outlier detection, we must first consider more deeply what constitutes an outlier, as well as how inlier and outlier features should be distributed. We begin by looking at a specific paradigm for thinking about not only outlier detection but classification in general, and then use this paradigm in order to develop a method for solving our specific problem of outlier detection using the HGR.

6.2.1 A Clustered View of the Universe

Ultimately, what defines an outlier comes down to how we define categories of "things". Inliers are "things we are interested in for this application" and outliers are "things we are not interested in for this application." Thus, a paradigm that allows us to consider all possible categories could shed light on how we can solve this problem.

One such paradigm is as follows:

> All things exist in clusters in some data space, and categories are defined as a cluster or group of clusters.

This is a view that comes out of the idea that classification is simply the process of identifying clusters of interest [4]. Under this paradigm, the purpose of classification in machine learning is to learn a representation that divides the clusters in one category from the clusters in another category in such a way that a decision boundary can easily be drawn between them.

Neural networks already make a similar assumption, assuming that the classes are separable and then attempting to learn representations that separate them so that a decision boundary can be drawn between them [14], and others are more explicit about the assumption that all data is clustered and learning is about separating out clusters [45]. If all categories that humans are interested in are clustered, then this problem reduces to attempting to find a lower-dimensional space where these clusters are still separated, so that we might draw a decision boundary between them.

6.2.2 Learning Clustered Features

We implement our outlier detector as an external module attached to an existing classification module. Thus, we seek to learn features $\Theta(X)$ from the data that are both predictive of Y and can be used to detect outliers.

If all our categories are clusters, then inliers are simply some group of clusters, and outliers are anything outside of these cluster centers. Indeed, many existing methods for outlier detection already assume this and leverage this assumption to find samples outside of the inlier clusters [21].

Our goal, then, is to learn features whereby inlier features are clustered tightly together, and known outlier features are far away from any cluster centers.

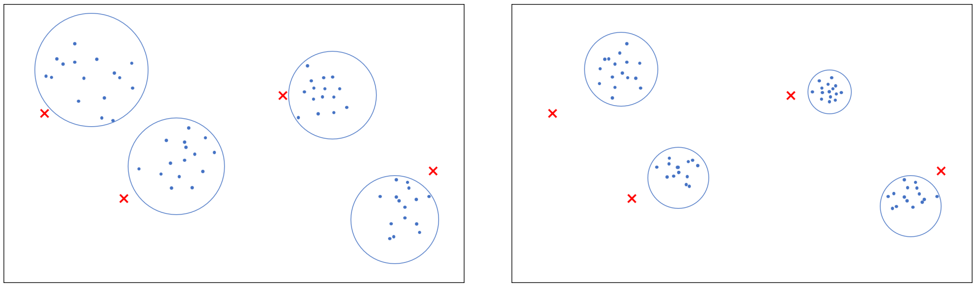


Figure 6-2: The more tightly clustered inlier samples (right) expose the outlier samples (labeled with red X) more readily than the case where inlier samples are not as tightly clustered (left).

We implement this by clustering the features using k-means clustering [14], assigning soft cluster labels D to all the samples, then applying the following optimization:

$$\begin{aligned} \min_{\substack{\Theta\colon \mathcal{X}\to\mathbb{R}^m \\ T\colon \mathbb{R}^m\to\mathcal{Y}}} & L(T(\Theta(X)), Y) - \\ & \lambda\big(\mathrm{HGR}_{\text{soft}}(\Theta(X_{ID}), D_{ID}) - \mathrm{HGR}_{\text{soft}}(\Theta(X_{OOD}), D_{OOD})\big). \end{aligned} \tag{6.1}$$

Where ID and OOD denote in-distribution (inlier) and out-of-distribution (outlier) samples, respectively. The training procedure is described in Algorithm 4.

This optimization has the effect of forcing a stronger correlation between the features and the clusters assigned to them, thus encouraging all samples in the same cluster to be located closer to the cluster center in features space. Meanwhile, the known outliers are pushed away from their nearest cluster center in each iteration. Thus, outlier samples are located far away from all clusters on inlier samples in feature space, making them easier to detect.

6.3 Experimental Setup

We test our method on a standard outlier detection image classification setup. Specifically, we train a network to classify one dataset, with a second training dataset used as the outlier dataset, and a third dataset used as a testing outlier dataset. We then extract an anomaly score, which we use to compute the AUC of outlier detection.

In this case, our anomaly score is the maximum softmax probability. This follows from the idea that our classifier will be more confident on samples whose representation in feature space are closer to that of other inlier samples [55].

In our experiment, we train our model on the training samples of CIFAR-10, with the validation samples of the LSUN dataset as the outlier training set, and set the test sets from SVHN and CIFAR-100 to be the test outlier sets. We compare to the outlier exposure method of Hendrycks et al. [55].[1]

[1] Code for these experiments can be found at `https://github.com/jklee-mit/outier_hgr`.

Algorithm 4 Learning features that isolate outliers

```
Data: Inlier samples $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ and outlier samples
      $\{(x_{ood,1}, y_{ood,1}), \ldots, (x_{ood,m}, y_{ood,m})\}$
Initialize networks $T$, $\Theta$ and $f$, for the main task and HGR feature extraction;
while not converge do
    Compute penultimate layer of $\Theta$ and denote as $z$;
    Collect $z_i$ for all training samples $x_i$;
    Cluster $z_i$ using $k$-means and get cluster labels $w$;
    Compute $g = E(f(z)|w)$ and orthogonalize;
    for batch in training set do
        Compute $f$ loss and backpropagate through $f$:
            $\text{hgr\_loss}(z, w) = -(E(f(z)g(w)) - 0.5tr(\text{cov}(f)\,\text{cov}(g)))$
        Compute $\Theta$ loss and backpropagate through $\Theta$ and $T$:
            $\text{loss} = L(T(\Theta(X)), Y) + \text{hgr\_loss}(z, w) - \text{hgr\_loss}(z_{ood}, w_{ood})$
    end
end
```

The CIFAR-10 and CIFAR-100 datasets[2] [66] are collections of color images drawn from 10 and 100 different categories of real-world subjects, respectively. These subjects consist of animate and inanimate objects, e.g., "ball," "fish," "bottle," "truck." Each dataset contains 50,000 training images and 10,000 test images, divided evenly among the classes.

The LSUN dataset[3] [130] is a scene recognition dataset that consists of 10 different environment labels (e.g., "bedroom", "bridge", "dining room"). There are ten million images in this dataset, and 3,000 in the validation set.

As stated previously, the SVHN dataset[4] [89] consists of labeled images of the digits 0 to 9 in the real world, obtained from house number images in Google Street View. The dataset contains 73,257 training samples and 26,032 testing samples.

Our network consists of a ResNet-18 model [54] (with initialization using pre-trained ImageNet weights) as the feature extractor, and we use 2-layer fully-connected

[2] `https://www.cs.toronto.edu/~kriz/cifar.html`

[3] `https://www.yf.io/p/lsun`

[4] `http://ufldl.stanford.edu/housenumbers/`

neural nets as our maximal correlation functions as well as our final classifier. We train the model for 100 epochs. For the clustering step, we use k-means clustering to assign cluster labels. We sweep through number of clusters from $k = 10$ to $k = 500$, eventually selecting $k = 50$ based on performance on the outlier validation sets (we sample from the training sets of the outlier datasets to form the validation sets).

Inlier Dataset - CIFAR-10

Outlier Training Dataset - LSUN

Outlier Test Dataset - SVHN

Outlier Test Dataset - CIFAR-100

Figure 6-3: Sample images for the outlier detection experiment

6.4 Experimental Results

Table 6.1 summarizes our outlier detection results. We can see that using the HGR-based method outperforms the outlier exposure method on the SVHN dataset, and

Table 6.1: Experimental results for outlier detection with AUC for each method in detecting the outlier samples. Values are reported with 95% confidence intervals over 5 trials.

Method	SVHN Detection	CIFAR-100 Detection
Baseline	51.6 ± 0.5	70.6 ± 1.2
Outlier Exposure	53.6 ± 0.5	73.8 ± 0.5
HGR	**77.4 ± 0.2**	**72.9 ± 0.4**

performs on par on the CIFAR-100 dataset. The CIFAR-100 detection problem is particularly difficult, as it draws its classes from the same superset of classes as CIFAR-10, while the SVHN samples represent a completely different type of image (digits vs. objects). Thus, our method appears to perform on par for difficult problems, but is also more adaptable to problems where the outliers are somewhat further away in some semantic sense from the inliers.

6.5 Concluding Remarks

In this chapter, we have introduced a new perspective on using the HGR to solve an old problem. Outlier detection illustrates a case where we have a secondary objective in our learning, but where the secondary objective is not necessarily defined by a tertiary variable provided to the system. In this case, by defining an auxiliary variable and applying the HGR principles, we can adapt to this setting as well.

Moving forwards, one interesting direction for exploration is to dig deeper into the very definition of an outlier. We have stated that what constitutes an anomaly is something that is defined by the system according to both the model and outlier detector used, as well as any examples of outlier samples. One could imagine that by modulating our definition of the auxiliary variable, different clusterings could be forced that include or exclude different inputs as being part of the outlier set. This flexibility could allow for a far more universal perspective on outlier detection.

There is also room here to explore the space of self-supervised tasks for devising an appropriate auxiliary variable. While clustering is one form of self-supervision

[111], other self-supervised tasks exist, each of which can learn a different set of labels [20, 99]. It follows then that different self-supervised tasks could define different types of clusterings for the data, which in turn would define new types of outliers. Whether these new definitions of outliers are useful or not remains to be seen, but with some careful selection of tasks, one could tune an outlier detector to obtain the separation desired between what is and is not considered a part of normal operations.

And if this theory of universal clustering holds true, then outlier detection is but one expression of an approach that could be used to model and solve all possible classification problems, if only the right way of reducing the size and complexity of the semantic space of the data can be found. This view of finding representations that highlight the boundary between categories could lead to novel algorithms for learning.

Chapter 7

Closing Thoughts

With the growth of automation and the proliferation of data science into every industry, new machine learning settings and concerns are constantly arising at a rapid pace. As such, the need to be adaptable and innovative in order to tackle these challenges is of utmost importance.

Time is also often of the essence in these settings. Not just runtime, which is always an important consideration, but the pace of development as well. New privacy laws can be devastating to the operation of existing systems, requiring rapid responses in order to ensure compliance. Rulings with respect to fairness also exhibit this same property. But even more concerning is that these systems will continue to be used as long as no laws prevent them, and thus, if private or fair solution cannot be deployed, then the system will simply continue to be allowed to harm the most vulnerable groups in our society.

This growth in machine learning also provides us with a wealth of opportunities, in addition to dangers, and the ability to rapidly adapt to a new dataset by using existing models, or to adjust learning in order to tackle secondary tasks such as outlier detection, can help maintain the momentum of development. Now, more than ever, the need for some useful grand perspective on learning is needed in order to provide a framework for this kind of thinking and enable this rapid growth.

The HGR maximal correlation is an important step in this process. By providing a universal perspective to view feature extraction and nonlinear correlation, it sets up a framework that allows us to solve a variety of problems with the right formulations. With the HGR, we can construct competitive algorithms to solve problems in fairness, privacy, transfer learning, and outlier detection. While this measure is not the best solution for every learning problem, it is nevertheless very adaptable and competitive with the state-of-the-art.

It is also important to reflect on the context of the HGR as well. This measure is over half a century old, and yet it continues to find use today, becoming more important as we use it in the context of modern machine learning. Many tools used in this field have roots reaching far back, and it is possible that by looking into the past, we may also find more useful ideas for the future.

Looking forward, developing this framework to additional applications is a clear direction. There are myriad fairness problems available, and while some problems seem straightforward to adapt to (such as fair clustering, which uses the demographic parity condition [23]), others, such as fair causal learning [8], may require some additional innovations to formulate the causal model in such a way that the HGR can be used to learn it fairly.

There is also a question to be asked here as to whether it is worth attempting to adapt the HGR to situations where it is not as well-suited (e.g., in the case of enforcing sufficiency), or whether the HGR simply represents one aspect of a larger paradigm that might be more generally adaptable.

With the outlier detection work, we have introduced the idea of using the HGR for an unsupervised task, though one in conjunction with a supervised task. Combining the problem of outlier detection with invariance learning could also yield methods that can be adapted to multiple domains [49]. Fully unsupervised learning via the HGR would be another direction to take to extend universality, as would self-supervised learning.

The ultimate goal would be to create a single, highly-adaptable framework that can be used to solve the increasing number of machine learning problems, by encoding side-information that informs the task at hand. Whether this is even possible on a grand scale or not remains to be seen, but as we continue to chip away at this problem, we will no doubt discover a treasure trove of tools and connections that can aid us in corralling the explosion in new machine learning tasks.

Appendix A

Notation and Glossary

$\mathbb{R}$	The set of real numbers
x, y, d	Scalars or vectors
X, Y, D	Random variables
$\hat{Y}, \hat{y}$	Prediction of Y or y, respectively
$\mathcal{X}, \mathcal{Y}, \mathcal{D}$	Alphabets
$\|\mathcal{X}\|$	Cardinality of $\mathcal{X}$
$\|x - y\|$	Absolute value of $x - y$
$\mathbf{B}$	Matrix
$\mathbf{I}$	Identity matrix
$f : \mathcal{X} \to \mathcal{Y}$	Function mapping $\mathcal{X}$ to $\mathcal{Y}$
$\mathcal{P}^{\mathcal{X}}$	Space of probability distributions over $\mathcal{X}$
$\|\|\mathbf{A}\|\|_F$	Frobenius norm of $\mathbf{A}$
$D_{\chi^2}(P\|\|Q)$	χ^2-divergence between P and Q
Φ, Ψ, Θ	Learned features (vectors or functions that output a vector)
$\boldsymbol{\Phi}, \boldsymbol{\Psi}, \boldsymbol{\Theta}$	Learned features (matrices or functions that output a matrix)

$\times$	Cartesian product
$\perp$	Independence
P	Probability distribution
$P(A)$	Probability of event A
μ	Mean
$\sup, \max, \min$	Supremum, maximum, and minimum
$\log()$	Logarithm with base e
$\exp()$	Exponential with base e
$\mathrm{cov}()$	Covariance matrix
$\mathbb{E}\left[\right]$	Expected value
$I(X;Y)$	Mutual information between X and Y
$\mathrm{tr}()$	Matrix trace
$\mathsf{relint}()$	Relative interior
HGR	Hirschfeld-Gebelein-Rényi maximal correlation
DEO	Difference in Equalized Opportunities
PPV	Positive predictive value (precision)

Appendix B

Selective Classification

B.1 Margin Distributions

One explanation for the phenomenon of bias in selective classification comes to us by way of margin distributions. The *margin* M of a classifier is defined as $\kappa(x)$ when $\hat{y}(x) = y$ and $-\kappa(x)$ otherwise. If we let τ be our threshold, then a selective classifier makes the correct prediction when $M(x) \geq \tau$ and incorrect predictions when $M(x) \leq -\tau$. We also denote its probability density function (PDF) and cumulative density function (CDF) as f_M and F_M, respectively. Then, the selective accuracy is

$$A_F(\tau) = \frac{1 - F_M(\tau)}{F_M(-\tau) + 1 - F_M(\tau)} \tag{B.1}$$

for a given threshold. We can analogously compute the selective precision by conditioning on $\hat{Y} = 1$,

$$PPV_F(\tau) = \frac{1 - F_{M|\hat{Y}=1}(\tau)}{F_{M|\hat{Y}=1}(-\tau) + 1 - F_{M|\hat{Y}=1}(\tau)}. \tag{B.2}$$

We can also analogously define the distributions of the margin for each group using $f_{M,d}$ and $F_{M,d}$ for group $d \in \mathcal{D}$.

Jones et al. [62] proposes a number of different situations for which average accuracy could increase but worst-group accuracy could decrease based on their relative

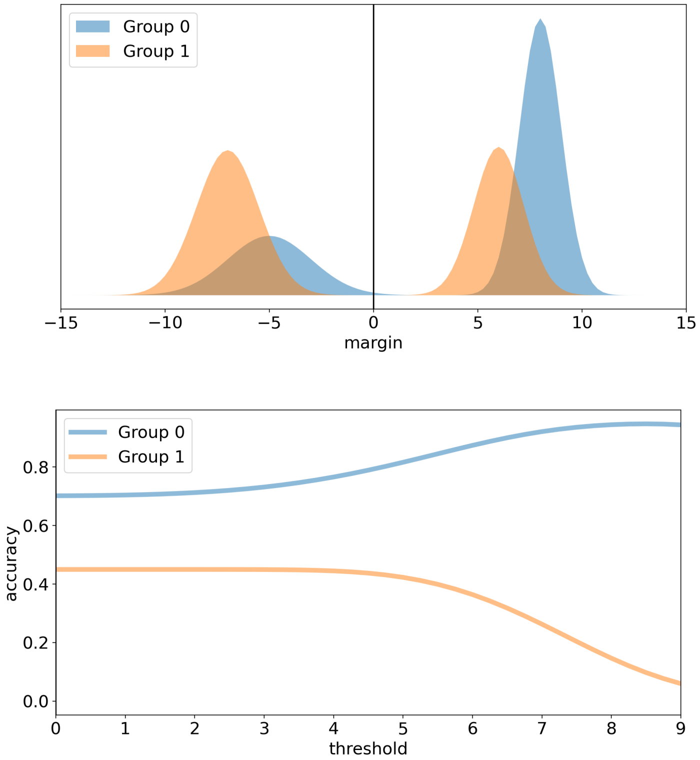


Figure B-1: (Top) When margin distributions are not aligned, (Bottom) then as we sweep over the threshold τ, the accuracies for the groups do not necessarily move in concert with one another.

margin distributions. For example, if F is left-log-concave (e.g., Gaussian), then $A_F(\tau)$ is monotonically increasing when $A_F(0) \geq 0.5$ and monotonically decreasing otherwise. Thus, if $A_F(0) > 0.5$ but $A_{F_d}(0) < 0.5$, then average accuracy may increase as we increase τ (and thus decrease coverage) but the accuracy on group d may decrease, thus resulting in magnified disparity. We can see this phenomenon illustrated in Figure B-1 This same phenomenon occurs with the precision when we condition on $\hat{Y} = 1$. In general, when margin distributions are not aligned between groups, disparity can increase as one sweeps over the threshold τ.

This margin-based view of selective classification will be important in our proof of Theorem 4.

B.2 Proof of Theorem 4

We first show that the accuracy $A_F(\tau)$ of a binary selective classification task is an increasing function if the confidence κ is constructed from a calibrated score function $R = s(x)$. The monotonicity of the precision $PPV_F(\tau)$ can be proved similarly.

The following lemma from [62] characterizes the condition for monotonicity of selective accuracy.

Lemma 5. *$A_F(\tau)$ is monotone increasing in τ if and only if*

$$\frac{f_M(\tau)}{f_M(-\tau)} \leq \frac{1 - F_M(\tau)}{F_M(-\tau)}, \tag{B.3}$$

for all $\tau \geq 0$.

Our proof also relies on the following lemma.

Lemma 6. *Suppose the score function R is calibrated by group, and that predictions are given by $\hat{Y} = \arg\max_{y \in \{0,1\}} P(Y = y|R = r)$. Denote the maximum a posteriori probability $S = \max\{R, 1 - R\}$, then*

$$P(Y = \hat{Y}|S = s, D = d) = s, \tag{B.4}$$

for all $d \in \mathcal{D}$.

Proof. Since R is calibrated by group, then $\forall a, b \in \mathcal{D}$,

$$P(Y = 1|R = r, D = a) = P(Y = 1|R = r, D = b) = r,$$

where $r \in [0, 1]$. Thus, for any $s \in [0.5, 1]$ and $d \in \mathcal{D}$, we have

$$\begin{aligned}
&P(Y = \hat{Y} | S = s, D = d) \\
&= P(Y = \hat{Y} | R \in \{s, 1-s\}, D = d) \\
&= P(Y = 1, \hat{Y} = 1 | R \in \{s, 1-s\}, D = d) \\
&\qquad + P(Y = 0, \hat{Y} = 0 | R \in \{s, 1-s\}, D = d) \\
&= \frac{P(Y = 1, \hat{Y} = 1, R \in \{s, 1-s\}, D = d)}{P(R \in \{s, 1-s\}, D = d)} \\
&\qquad + \frac{P(Y = 0, \hat{Y} = 0, R \in \{s, 1-s\}, D = d)}{P(R \in \{s, 1-s\}, D = d)} \\
&\overset{(a)}{=} \frac{P(Y = 1, \hat{Y} = 1, R = s, D = d)}{P(R \in \{s, 1-s\}, D = d)} \\
&\qquad + \frac{P(Y = 0, \hat{Y} = 0, R = 1-s, D = d)}{P(R \in \{s, 1-s\}, D = d)} \\
&= \frac{P(Y = 1 | R = s, D = d) P(R = s, D = d)}{P(R = s, D = d) + P(R = 1-s, D = d)} \\
&\quad + \frac{P(Y = 0 | R = 1-s, D = d) P(R = 1-s, D = d)}{P(R = s, D = d) + P(R = 1-s, D = d)} \\
&\overset{(b)}{=} s,
\end{aligned}$$

where (a) follows from the fact that $\hat{Y} = 1$ iff $R \geq 0.5$, and $\hat{Y} = 0$ iff $R < 0.5$, and (b) is due to the calibration by group assumption $P(Y = 0 | R = 1-s, D = d) = s$. □

By Lemma 6, the accuracy $P(Y = \hat{Y} | S = s, D = d)$ is independent of the group D given S and we can drop the conditioning of the group in the following proof.

In the selective classification problem, we convert the maximum a posteriori probability s into confidence κ using

$$\kappa(s) = \frac{1}{2} \log \left(\frac{s}{1-s} \right), \tag{B.5}$$

which maps $[0.5, 1]$ to $[0, \infty]$. So for any sample with confidence $z \in \mathbb{R}^+$,

$$\begin{aligned}
P(Y = \hat{Y} | \kappa = z) &= P(Y = \hat{Y} | S = \kappa^{-1}(z)) \\
&= \kappa^{-1}(z),
\end{aligned} \tag{B.6}$$

where $\kappa^{-1}(\cdot)$ is the inverse function of $\kappa(\cdot)$. We use $f_\kappa(z)$ to denote the PDF of the confidence score κ for $z \in \mathbb{R}^+$, and then the PDF of the margin $f_M(t)$ can be written as,

$$f_M(t) = \begin{cases} P(Y = \hat{Y}|\kappa = t) f_\kappa(t), & \text{for } t \geq 0 \\ P(Y \neq \hat{Y}|\kappa = -t) f_\kappa(-t), & \text{for } t < 0, \end{cases}$$

or equivalently,

$$f_M(t) = \begin{cases} \kappa^{-1}(t) f_\kappa(t), & \text{for } t \geq 0 \\ (1 - \kappa^{-1}(-t)) f_\kappa(-t), & \text{for } t < 0. \end{cases} \tag{B.7}$$

It can be verified that $\kappa^{-1}(z)$ is a increasing function for $z \in \mathbb{R}^+$, and $\kappa^{-1}(0) = \frac{1}{2}$. Thus,

$$\frac{f_M(z)}{f_M(-z)} = \frac{\kappa^{-1}(z) f_\kappa(z)}{(1 - \kappa^{-1}(z)) f_\kappa(z)} = \frac{\kappa^{-1}(z)}{(1 - \kappa^{-1}(z))} \geq 1. \tag{B.8}$$

We can conclude that the CDF of the margin $F_M(t)$ satisfies

$$F_M(0) = \int_{-\infty}^{0} f_M(t) dt < \frac{1}{2}, \tag{B.9}$$

which implies that $A_F(0) > 0.5$.

To show that $A_F(\tau)$ is monotonically increasing with the threshold τ, we need to verify the condition in Lemma 5. Note that

$$\begin{aligned} \frac{1 - F_M(\tau)}{F_M(-\tau)} &= \frac{\int_\tau^\infty f_M(t) dt}{\int_{-\infty}^{-\tau} f_M(t) dt} \\ &= \frac{\int_\tau^\infty \kappa^{-1}(t) f_\kappa(t) dt}{\int_\tau^\infty (1 - \kappa^{-1}(t)) f_\kappa(t) dt} \\ &\geq \frac{\kappa^{-1}(\tau) \int_\tau^\infty f_\kappa(t) dt}{(1 - \kappa^{-1}(\tau)) \int_\tau^\infty f_\kappa(t) dt} \\ &= \frac{f_M(\tau)}{f_M(-\tau)}, \end{aligned}$$

which completes the proof for the selective accuracy.

By replacing the margin distribution $f_M(t)$ with the margin distribution condition on $\hat{Y} = 1$, i.e., $f_{M|\hat{Y}=1}(t)$, the monotonicity of the precision $PPV_F(\tau)$ can be obtained following similar steps.

Note that the condition for monotonicity of the precision is given by

$$\frac{f_{M|\hat{Y}=1}(\tau)}{f_{M|\hat{Y}=1}(-\tau)} \leq \frac{1 - F_{M|\hat{Y}=1}(\tau)}{F_{M|\hat{Y}=1}(-\tau)}, \tag{B.10}$$

and Lemma 6 is replaced by the following simple fact due to calibration by group

$$\begin{aligned} & P(Y = 1|\hat{Y} = 1, S = s) \\ & = P(Y = 1|R = s) \\ & = s. \end{aligned} \tag{B.11}$$

In our proof, it only requires that the confidence function κ is a increasing function that maps $[0.5, 1]$ to $[0, \infty]$, so that $\kappa^{-1}(\cdot)$ is a increasing function and $\kappa^{-1}(0) = \frac{1}{2}$. Thus, Theorem 4 also holds for confidence functions satisfying these conditions, which is not limited to the function in (5.3).

B.3 Mutual Information Upper Bound

In order to compare the HGR method with another method rooted in mutual information, we derive an algorithm based on a novel upper bound of the mutual information which is well-suited for this application. Existing works using mutual information for fairness are ill-equipped to handle the sufficiency condition, as they assume that it is not the features that will be conditioned on, but rather that the penalty will be the mutual information between the sensitive attribute and the features (e.g., penalizing $I(U; D)$ for demographic parity), possibly conditioned on the label (e.g., penalizing $I(U; D|Y)$ in the case of equalized opportunities). As such, existing methods either assume that the variable being conditioned on is discrete [19, 53, 132], become unstable when the features are placed in the condition [86], or simply do not allow for conditioning of this type due to their formulation [46, 6].

Thus, in order to approximate the mutual information for our purposes, we must first derive an upper bound for the mutual information which is computable in our applications. Our bound is inspired by the work of [22] and is stated in the following theorem:

Theorem 7. *For random variables X, Y and Z, we have*

$$I_{\mathrm{UB}}(X;Y|Z) \geq I(X;Y|Z), \tag{B.12}$$

where equality is achieved if and only if $X \perp Y \mid Z$, and

$$\begin{aligned} I_{\mathrm{UB}}(X;Y|Z) \triangleq\, & \mathbb{E}_{P_{X,Y,Z}}\left[\log P(Y|X,Z)\right] \\ & - \mathbb{E}_{P_X}\left[\mathbb{E}_{P_{Y,Z}}\left[\log P(Y|X,Z)\right]\right]. \end{aligned} \tag{B.13}$$

Proof. The conditional mutual information can be written as

$$\begin{aligned} & I(X;Y|Z) \\ & = \mathbb{E}_{P_{X,Y,Z}}\left[\log P(Y|X,Z)\right] - \mathbb{E}_{P_{Y,Z}}\left[\log P(Y|Z)\right]. \end{aligned} \tag{B.14}$$

Thus,

$$\begin{aligned} & I_{\mathrm{UB}}(X;Y|Z) - I(X;Y|Z) \\ & = \mathbb{E}_{P_{Y,Z}}\left[\log P(Y|Z) + \mathbb{E}_{P_X}\left[-\log P(Y|X,Z)\right]\right]. \end{aligned} \tag{B.15}$$

Note that $-\log(\cdot)$ is convex,

$$\begin{aligned} \mathbb{E}_{P_X}\left[-\log P(Y|X,Z)\right] & \geq -\log \mathbb{E}_{P_X}\left[P(Y|X,Z)\right] \\ & = -\log P(Y|Z), \end{aligned} \tag{B.16}$$

which completes the proof. □

Thus, $I(Y;D|U)$ can be upper bounded by I_{UB} as:

$$\begin{aligned} I(Y;D|U) \leq \mathbb{E}_{P_{X,Y,D}}\left[\log P(Y|\Theta(X),D)\right] \\ - \mathbb{E}_{P_D}\left[\mathbb{E}_{P_{X,Y}}\left[\log P(Y|\Theta(X),D)\right]\right]. \end{aligned} \tag{B.17}$$

Since $P(y|\Theta(x),d)$ is unknown in practice, we need to use a variational distribution $q(y|\Theta(x),d;\theta)$ with parameter θ to approximate it. Here, we adopt a neural net that predicts Y based on feature $\Theta(X)$ and sensitive attribute D as our variational model $q(y|\Theta(x),d;\theta)$.

Algorithm 5 Training with sufficiency-based regularizer

```
Data: Training samples $\{(x_1,y_1,d_1),\ldots,(x_n,y_n,d_n)\}$, $\{\widetilde{d}_1,\ldots,\widetilde{d}_n\}$, which are drawn
      i.i.d. from the empirical distribution $\hat{P}_D$
Initialize $\Theta$, $T$ (parameterized by $\theta_\Theta$ and $\theta_T$, respectively) and $\theta_d$ with pre-trained
  model, and let $n_d$ be the number of samples in group $d$.
Compute the following losses:
Group-specific losses $L_d = -\sum_{i:\ d_i=d} \log q(y_i|\Theta(x_i);\theta)$
Joint loss $L_0 = \frac{1}{n}\sum_{i=1}^{n} L\big(T(\Theta(x_i)),y_i\big)$
Regularizer loss $L_R$ defined in (B.18) including both Group-specific loss and Group-
  agnostic loss
for each training iteration do
    for $d = 1,\ldots,|\mathcal{D}|$ do // Fit group-specific models
        for $j = 1,\ldots,M$ do // For each batch
            $\theta_d \leftarrow \theta_d - \frac{1}{n_d}\eta_d \nabla_\theta L_d$
        end
    end
    for $j = 1,\ldots,N$ do // For each batch
        $\theta_\Theta \leftarrow \theta_\Theta - \frac{1}{n}\eta_f \nabla_{\theta_\Theta}(L_0 + \lambda L_R)$ // Update feature extractor
        $\theta_T \leftarrow \theta_T - \frac{1}{n}\eta \nabla_{\theta_T} L_0$ // Update joint classifier
    end
end
```

However, in many cases, X will be continuous, high-dimensional data (e.g., images), while D will be a discrete, categorical variable (e.g., gender, ethnicity). Therefore, it would be more convenient to instead formulate the model as $q(y|\Theta(x);\theta_d)$, i.e., to train a *group-specific* model for each $d \in \mathcal{D}$ to approximate $P(y|\Theta(x),d)$, instead of treating D as a single input to the neural net.

Then, we can compute the first term of the upper bound as the negative cross-entropy of the training samples using the "correct" classifier for each group (group-specific loss), and the second term as the cross-entropy of the samples using a randomly selected classifier (group-agnostic loss) drawn according to the marginal distribution P_D. Thus, by replacing all expectations in (B.17) with empirical averages, the regularizer is given by

$$L_R \triangleq \frac{1}{n} \sum_{i=1}^{n} \Big(\log q(y_i | \Theta(x_i); \theta_{d_i}) - \log q(y_i | \Theta(x_i); \theta_{\widetilde{d_i}}) \Big), \tag{B.18}$$

where $\widetilde{d_i}$ are drawn i.i.d. from the marginal distribution P_D, and for $d \in \mathcal{D}$,

$$\theta_d = \arg\max_{\theta} \sum_{i:\ d_i = d} \log q(y_i | \Theta(x_i); \theta). \tag{B.19}$$

Let T denote a *joint classifier* over all groups which is used to make final predictions, such that $\hat{y} = T(\Theta(x))$, then the overall loss function is

$$\begin{aligned} \min_{\theta_T, \theta_\Theta} \frac{1}{n} \sum_{i=1}^{n} \Big(& L\big(T(\Theta(x_i)), y_i\big) + \lambda \log q(y_i | \Theta(x_i); \theta_{d_i}) \\ & - \lambda \log q(y_i | \Theta(x_i); \theta_{\widetilde{d_i}}) \Big). \end{aligned} \tag{B.20}$$

In practice, we train our model by alternating between the fitting steps in (B.19) and feature updating steps in (B.20), and the overall training process is described in Algorithm 5 and Figure B-2. Intuitively, by trying to minimize the difference between the log-probability of the output of the correct model and that of the randomly-chosen one, we are trying to enforce $\Theta(x)$ to have the property that all group-specific models trained on it will be the same; that is:

$$q(y | \Theta(x); \theta_a) = q(y | \Theta(x); \theta_b), \quad \forall a, b \in \mathcal{D}. \tag{B.21}$$

This happens when $P(Y | \Theta(X), D) = P(Y | \Theta(X))$, which implies the sufficiency condition $Y \perp D | U$.

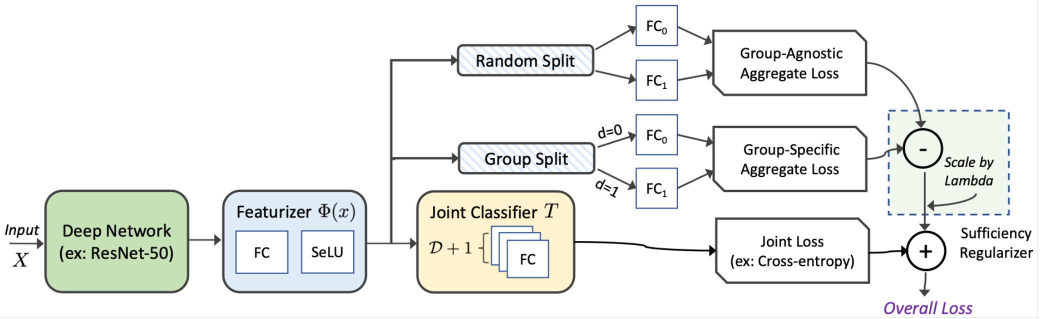


Figure B-2: Diagram illustrating the computation of our sufficiency-based loss when D is binary.

Bibliography

[1] Jamal Alasadi, Ahmed Al Hilli, and Vivek K Singh. Toward fairness in face matching algorithms. In *Proceedings of the 1st International Workshop on Fairness, Accountability, and Transparency in MultiMedia*, pages 19–25, 2019.

[2] Zahir Alsulaimawi. Variational bound of mutual information for fairness in classification. In *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–6. IEEE, 2020.

[3] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: There's software used across the country to predict future criminals. And it's biased against blacks. *ProPublica*, 2016.

[4] Phips Arabie, Larry Hubert, and Geert De Soete. *Clustering and classification.* World Scientific, 1996.

[5] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

[6] Sina Baharlouei, Maher Nouiehed, Ahmad Beirami, and Meisam Razaviyayn. Rènyi fair inference. *arXiv preprint arXiv:1906.12005*, 2019.

[7] Yajie Bao, Yang Li, Shao-Lun Huang, Lin Zhang, Amir R. Zamir, and Leonidas J. Guibas. An information-theoretic metric of transferability for task transfer learning. `https://openreview.net/forum?id=BkxAUjRqY7`, 2019.

[8] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning.* fairmlbook.org, 2019. `http://www.fairmlbook.org`.

[9] Rachel KE Bellamy, Kuntal Dey, Michael Hind, Samuel C Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilović, et al. AI fairness 360: An extensible toolkit for detecting and mitigating algorithmic bias. *IBM Journal of Research and Development*, 63(4/5):4–1, 2019.

[10] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in Neural Information Processing Systems*, pages 137–144, 2007.

[11] Yoshua Bengio. Time to rethink the publication process in machine learning, Mar 2020.

[12] Yoshua Bengio, Richard Janda, Yun William Yu, Daphne Ippolito, Max Jarvie, Dan Pilat, Brooke Struck, Sekoul Krastev, and Abhinav Sharma. The need for privacy with public digital contact tracing during the COVID-19 pandemic. *The Lancet Digital Health*, 2(7):e342–e344, 2020.

[13] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4183–4192, 2020.

[14] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[15] Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion Proceedings of The 2019 World Wide Web Conference*, 2019.

[16] Leo Breiman and Jerome H. Friedman. Estimating optimal transformations for multiple regression and correlation. *J. Am. Stat. Assoc.*, 80(391):580–598, September 1985.

[17] Lyle D Broemeling. An account of early statistical inference in Arab cryptology. *The American Statistician*, 65(4):255–257, 2011.

[18] Yuheng Bu, Tony Wang, and Gregory W Wornell. SDP methods for sensitivity-constrained privacy funnel and information bottleneck problems. In *2021 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2021.

[19] Flavio Calmon, Dennis Wei, Bhanukiran Vinzamuri, Karthikeyan Natesan Ramamurthy, and Kush R Varshney. Optimized pre-processing for discrimination prevention. In *Advances in Neural Information Processing Systems*, pages 3992–4001, 2017.

[20] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2229–2238, 2019.

[21] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[22] Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. CLUB: A contrastive log-ratio upper bound of mutual information. In *International Conference on Machine Learning*, pages 1779–1788. PMLR, 2020.

[23] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. Fair clustering through fairlets. *arXiv preprint arXiv:1802.05733*, 2018.

[24] Jaewoong Cho, Gyeongjo Hwang, and Changho Suh. A fair classifier using mutual information. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 2521–2526. IEEE, 2020.

[25] Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data*, 5(2):153–163, 2017.

[26] U.S. Equal Employment Opportunity Commission. Department of labor, & Department of justice.(1978). Uniform guidelines on employee selection procedures. *Federal Register*, 1978.

[27] Glinda S. Cooper and Vanessa Meterko. Cognitive bias research in forensic science: A systematic review. *Forensic Science International*, 297:35–46, 2019.

[28] Sam Corbett-Davies and Sharad Goel. The measure and mismeasure of fairness: A critical review of fair machine learning. *arXiv preprint arXiv:1808.00023*, 2018.

[29] Corinna Cortes, Giulia DeSalvo, and Mehryar Mohri. Learning with rejection. In *International Conference on Algorithmic Learning Theory*, pages 67–82. Springer, 2016.

[30] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006.

[31] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[32] Jinshuo Dong, Aaron Roth, and Weijie J Su. Gaussian differential privacy. *arXiv preprint arXiv:1905.02383*, 2019.

[33] Dheeru Dua and Casey Graff. UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences, 2017.

[34] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[35] Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1657–1664, 2013.

[36] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proc. Int. Conf. Machine Learning (ICML)*, volume 70, pages 1126–1135, 2017.

[37] Vojtech Franc and Daniel Prusa. On discriminative learning of prediction uncertainty. In *International Conference on Machine Learning*, pages 1963–1971. PMLR, 2019.

[38] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proc. ACM SIGSAC Conf. Computer, Communications Security*, pages 1322–1333, 2015.

[39] Weihao Gao, Sewoong Oh, and Pramod Viswanath. Demystifying fixed k-nearest neighbor information estimators. *IEEE Transactions on Information Theory*, 64(8):5629–5661, 2018.

[40] Hans Gebelein. Das statistische problem der korrelation als variations-und eigenwertproblem und sein zusammenhang mit der ausgleichsrechnung. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 21(6):364–379, 1941.

[41] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, pages 4878–4887, 2017.

[42] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. On deep learning with label differential privacy. *arXiv preprint arXiv:2102.06062*, 2021.

[43] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2551–2559, 2015.

[44] Soumya Ghosh, Q. Vera Liao, Karthikeyan Natesan Ramamurthy, Jiri Navratil, Prasanna Sattigeri, Kush R. Varshney, and Yunfeng Zhang. Uncertainty quantification 360: A holistic toolkit for quantifying and communicating the uncertainty of AI, 2021.

[45] Sebastian Goebl, Xiao He, Claudia Plant, and Christian Böhm. Finding the optimal subspace for clustering. In *2014 IEEE International Conference on Data Mining*, pages 130–139, 2014.

[46] Vincent Grari, Boris Ruf, Sylvain Lamprier, and Marcin Detyniecki. Fairness-aware neural Rényi minimization for continuous features. *arXiv preprint arXiv:1911.04929*, 2019.

[47] Nina Grgic-Hlaca, Muhammad Bilal Zafar, Krishna P Gummadi, and Adrian Weller. The case for process fairness in learning: Feature selection for fair decision making. In *NIPS Symposium on Machine Learning and the Law*, volume 1, page 2, 2016.

[48] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

[49] Ruocheng Guo, Pengchuan Zhang, Hao Liu, and Emre Kiciman. Out-of-distribution prediction with invariant risk minimization: The limitation and an effective fix. *arXiv preprint arXiv:2101.07732*, 2021.

[50] Rakesh Gupta and Lev-Arie Ratinov. Text categorization with knowledge transfer from heterogeneous data sources. In *Proceedings of the National Conference on Artificial Intelligence*, volume 2, pages 842–847, 01 2008.

[51] Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *Proc. Int. Conf. Machine Learning (ICML)*, pages 1737–1746, 2015.

[52] Mandana Hamidi and Ali Borji. Invariance analysis of modified C2 features: case study—handwritten digit recognition. *Machine Vision and Applications*, 21(6):969–979, 2010.

[53] Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems 29*, pages 3315–3323, Barcelona, Spain, December 2016.

[54] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[55] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep anomaly detection with outlier exposure. *arXiv preprint arXiv:1812.04606*, 2018.

[56] Hermann O. Hirschfeld. A connection between correlation and contingency. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31(4):520–524, 1935.

[57] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[58] Shao-Lun Huang, Anuran Makur, Gregory W Wornell, and Lizhong Zheng. On universal features for high-dimensional learning and inference. *arXiv preprint arXiv:1911.09105*, 2019.

[59] Shao-Lun Huang, Anuran Makur, Lizhong Zheng, and Gregory W Wornell. An information-theoretic approach to universal feature selection in high-dimensional inference. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 1336–1340. IEEE, 2017.

[60] Shao-Lun Huang and Xiangxiang Xu. On the sample complexity of HGR maximal correlation functions for large datasets. *IEEE Transactions on Information Theory*, 2020.

[61] Shao-Lun Huang and Lizhong Zheng. Linear information coupling problems. In *2012 IEEE International Symposium on Information Theory Proceedings*, pages 1029–1033. IEEE, 2012.

[62] Erik Jones, Shiori Sagawa, Pang Wei Koh, Ananya Kumar, and Percy Liang. Selective classification can magnify disparities across groups. *arXiv preprint arXiv:2010.14134*, 2020.

[63] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, 2011.

[64] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. *arXiv preprint arXiv:2012.07421*, 2020.

[65] Ron Kohavi. Scaling up the accuracy of naïve-Bayes classifiers: A decision-tree hybrid. In *KDD*, volume 96, pages 202–207, 1996.

[66] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, Toronto, Canada, 2009.

[67] Max Kuhn, Kjell Johnson, et al. *Applied predictive modeling*, volume 26. Springer, 2013.

[68] Gukyeong Kwon, Mohit Prabhushankar, Dogancan Temel, and Ghassan AlRegib. Backpropagated gradient representations for anomaly detection. *arXiv preprint arXiv:2007.09507*, 2020.

[69] H. O. Lancaster. The structure of bivariate distributions. *The Annals of Mathematical Statistics*, 29(3):719–736, 1958.

[70] Nicholas D. Lane, Sourav Bhattacharya, Petko Georgiev, Claudio Forlivesi, Lei Jiao, Lorena Qendro, and Fahim Kawsar. Deepx: A software accelerator for low-power deep learning inference on mobile devices. In *Proc. Int. Conf. Information Processing in Sensor Networks*, page 23, 2016.

[71] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

[72] Adrien Marie Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes: avec un supplément contenant divers perfectionnemens de ces méthodes et leur application aux deux comètes de 1805*. Courcier, 1806.

[73] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5542–5550, 2017.

[74] Fei-Fei Li, Andrej Karpathy, and Justin Johnson. Tiny imagenet visual recognition challenge. `https://tiny-imagenet.herokuapp.com/`, 2015. [Online; accessed 13-May-2019].

[75] Ruizhe Li, Xiao Li, Chenghua Lin, Matthew Collinson, and Rui Mao. A stable variational autoencoder for text modelling. *arXiv preprint arXiv:1911.05343*, 2019.

[76] Chee Peng Lim and Robert F. Harrison. Online pattern classification with multiple neural network systems: an experimental study. *IEEE Trans. Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 33(2):235–247, 2003.

[77] Zachary C Lipton and Jacob Steinhardt. Troubling trends in machine learning scholarship: Some ML papers suffer from flaws that could mislead the public and stymie future research. *Queue*, 17(1):45–77, 2019.

[78] Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. Deep multi-task learning with shared memory. *CoRR*, abs/1609.07222, 2016. `http://arxiv.org/abs/1609.07222`.

[79] Weitang Liu, Xiaoyun Wang, John D Owens, and Yixuan Li. Energy-based out-of-distribution detection. *arXiv preprint arXiv:2010.03759*, 2020.

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

[81] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. *CoRR*, abs/1502.02791, 2015. `http://arxiv.org/abs/1502.02791`.

[82] Milan Lopuhaä-Zwakenberg. The privacy funnel from the viewpoint of local differential privacy. *arXiv preprint arXiv:2002.01501*, 2020.

[83] Ali Makhdoumi, Salman Salamatian, Nadia Fawaz, and Muriel Médard. From the information bottleneck to the privacy funnel. In *2014 IEEE Information Theory Workshop (ITW 2014)*, pages 501–505. IEEE, 2014.

[84] Anuran Makur, Fabián Kozynski, Shao-Lun Huang, and Lizhong Zheng. An efficient algorithm for information decomposition and extraction. In *Proc. Allerton Conf. Commun., Control, Computing*, pages 972–979, Monticello, IL, 2015.

[85] Giuseppe Manco and Giuseppe Pirrò. Differential privacy and neural networks: A preliminary analysis. In *Proc. Int. Workshop Personal Analytics, Privacy*, pages 23–35, 2017.

[86] Jérémie Mary, Clément Calauzenes, and Noureddine El Karoui. Fairness-aware learning for continuous attributes and treatments. In *International Conference on Machine Learning*, pages 4382–4391, 2019.

[87] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. Meta-learning with temporal convolutions. *CoRR*, abs/1707.03141, 2017. `http://arxiv.org/abs/1707.03141`.

[88] Daniel Moyer, Shuyang Gao, Rob Brekelmans, Aram Galstyan, and Greg Ver Steeg. Invariant representations without adversarial training. In *Advances in Neural Information Processing Systems*, pages 9084–9093, 2018.

[89] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[90] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999, 2018. `http://arxiv.org/abs/1803.02999`.

[91] Northpointe. Practitioner's guide to COMPAS core. *equivant*, 2015.

[92] United States. Bureau of the Census, Inter university Consortium for Political, and Social Research. *Census of Population and Housing, 1990 (United States).: Summary tape file 1A*, volume 9575. Inter-university Consortium for Political and Social Research, 1992.

[93] Paul Ohm. Broken promises of privacy: Responding to the surprising failure of anonymization. *UCLA L. Rev.*, 57:1701, 2009.

[94] Kaoru Ota, Minh Son Dao, Vasileios Mezaris, and Francesco G. B. De Natale. Deep learning for mobile multimedia: A survey. *ACM Trans. Multimedia Computing, Communications, and Applications*, 13(3s):34, 2017.

[95] Sinno Jialin Pan, James T. Kwok, and Qiang Yang. Transfer learning via dimensionality reduction. In *Proc. AAAI Conf. Artificial Intelligence*, volume 8, pages 677–682, 2008.

[96] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *Autodiff Workshop, Conf. Neural Information Processing Systems*, Long Beach, CA, 2017.

[97] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1406–1415, 2019.

[98] Claudio Persello and Lorenzo Bruzzone. Kernel-based domain-invariant feature selection in hyperspectral images for transfer learning. *IEEE transactions on geoscience and remote sensing*, 54(5):2615–2626, 2015.

[99] AJ Piergiovanni, Anelia Angelova, and Michael S Ryoo. Evolving losses for unsupervised video representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 133–142, 2020.

[100] ProPublica. COMPAS recidivism risk score data and analysis. `https://www.propublica.org/datastore/dataset/compas-recidivism-risk-score-data-and-analysis`, 2021.

[101] David Qiu. *Representation and transfer learning using information-theoretic approximations.* PhD thesis, Massachusetts Institute of Technology, 2020.

[102] Michael Redmond and Alok Baveja. A data-driven software tool for enabling cooperative information sharing among police departments. *European Journal of Operational Research*, 141(3):660–678, 2002.

[103] Alfréd Rényi. On measures of dependence. *Acta Mathematica Hungarica*, 10(3-4):441–451, 1959.

[104] Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. The risks of invariant risk minimization. *arXiv preprint arXiv:2010.05761*, 2020.

[105] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016. `http://arxiv.org/abs/1606.04671`.

[106] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[107] Salman Salamatian, Flavio du Pin Calmon, Nadia Fawaz, Ali Makhdoumi, and Muriel Médard. Privacy-utility tradeoff and privacy funnel. *Unpublished preprint,* ***http: // www. mit. edu/∼salmansa/ files/ privacy_ TIFS. pdf***, 2020.

[108] Salman Salamatian, Amy Zhang, Flavio du Pin Calmon, Sandilya Bhamidipati, Nadia Fawaz, Branislav Kveton, Pedro Oliveira, and Nina Taft. Managing your private and public data: Bringing down inference attacks against your privacy. *IEEE Journal of Selected Topics in Signal Processing*, 9(7):1240–1255, 2015.

[109] Andrew D Selbst, Danah Boyd, Sorelle A Friedler, Suresh Venkatasubramanian, and Janet Vertesi. Fairness and abstraction in sociotechnical systems. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 59–68, 2019.

[110] Jiaming Song and Stefano Ermon. Understanding the limitations of variational mutual information estimators. *arXiv preprint arXiv:1910.06222*, 2019.

[111] Yu Sun, Eric Tzeng, Trevor Darrell, and Alexei A Efros. Unsupervised domain adaptation through self-supervision. *arXiv preprint arXiv:1909.11825*, 2019.

[112] Bahar Taskesen, Viet Anh Nguyen, Daniel Kuhn, and Jose Blanchet. A distributionally robust approach to fair classification. *arXiv preprint arXiv:2007.09530*, 2020.

[113] Ardhendu Tripathy, Ye Wang, and Prakash Ishwar. Privacy-preserving adversarial networks. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 495–505, 2019.

[114] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962*, 2019.

[115] Selen Uguroglu and Jaime Carbonell. Feature selection for transfer learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 430–442. Springer, 2011.

[116] Charles F. Van Loan and Gene H. Golub. *Matrix computations.* The Johns Hopkins University Press, 1996.

[117] Rocio Vargas, Amir Mosavi, and Ramon Ruiz. Deep learning: A review. Working paper, QUT Library, January 2017.

[118] Sahil Verma and Julia Rubin. Fairness definitions explained. In *2018 IEEE/ACM International Workshop on Software Fairness (FairWare)*, pages 1–7, 2018.

[119] Lichen Wang, Jiaxiang Wu, Shao-Lun Huang, Lizhong Zheng, Xiangxiang Xu, Lin Zhang, and Junzhou Huang. An efficient approach to informative feature extraction from multimodal data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5281–5288, 2019.

[120] Zeyu Wang, Klint Qinami, Ioannis Christos Karakozis, Kyle Genova, Prem Nair, Kenji Hata, and Olga Russakovsky. Towards fairness in visual recognition: Effective strategies for bias mitigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8919–8928, 2020.

[121] Zhipeng Wang and David W Scott. Nonparametric density estimation for high-dimensional data—algorithms and applications. *Wiley Interdisciplinary Reviews: Computational Statistics*, 11(4):e1461, 2019.

[122] Larry Wasserman. *All of statistics: a concise course in statistical inference.* Springer Science & Business Media, USA, 2013.

[123] Jim Winkens, Rudy Bunel, Abhijit Guha Roy, Robert Stanforth, Vivek Natarajan, Joseph R Ledsam, Patricia MacWilliams, Pushmeet Kohli, Alan Karthikesalingam, Simon Kohl, et al. Contrastive training for improved out-of-distribution detection. *arXiv preprint arXiv:2007.05566*, 2020.

[124] Tian Xu, Jennifer White, Sinan Kalkan, and Hatice Gunes. Investigating bias and fairness in facial expression recognition. In *European Conference on Computer Vision*, pages 506–523. Springer, 2020.

[125] Ke Yang and Julia Stoyanovich. Measuring fairness in ranked outputs. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, pages 1–6, 2017.

[126] Kyra Yee, Uthaipon Tantipongpipat, and Shubhanshu Mishra. Image cropping on twitter: Fairness metrics, their limitations, and the importance of representation, design, and agency. *arXiv preprint arXiv:2105.08667*, 2021.

[127] Mehmet Yigit Yildirim, Mert Ozer, and Hasan Davulcu. Leveraging uncertainty in deep learning for selective classification. *arXiv preprint arXiv:1905.09509*, 2019.

[128] Ed Yong. A popular algorithm is no better at predicting crimes than random people, Jan 2018.

[129] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, pages 3320–3328, 2014.

[130] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[131] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rogriguez, and Krishna P Gummadi. Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pages 962–970. PMLR, 2017.

[132] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *International Conference on Machine Learning*, pages 325–333, 2013.

[133] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 335–340, 2018.

[134] Daniel Zhang, Saurabh Mishra, Erik Brynjolfsson, John Etchemendy, Deep Ganguli, Barbara Grosz, Terah Lyons, James Manyika, Juan Carlos Niebles, Michael Sellitto, et al. The AI index 2021 annual report. *arXiv preprint arXiv:2103.06312*, 2021.