ERASMUS UNIVERSITY ROTTERDAM

MASTER THESIS

Controlling and Mitigating Biases with Adversarial Ensemble-based Methods

---

Author:
Nicholas SMEELE

Supervisor:
Prof. Dr. Bas DONKERS

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Data Science and Marketing Analytics
Erasmus School of Economics

October 30, 2020

ERASMUS UNIVERSITY ROTTERDAM

Abstract

Departments of Econometrics and Marketing

Erasmus School of Economics

Master of Science

Controlling and Mitigating Biases with Adversarial Ensemble-based Methods

by Nicholas SMEELE

Machine learning has become an integral part of society and business. Although powerful, well-trained models can, even in the absence of intent, reflect and accentuate harmful biases that are present in the training data. This thesis presents an adversarial ensemble-based framework for mitigating unintended biases while controlling for acceptable discrimination through admissible information. The input to the network is segregated into two model components, here Mitigation Expert (ME) and Control Expert (CE), and ensembled into the main task predictor. By introducing an adversarial game between the predictor and adversary, the unintended gender bias is mitigated from ME. The CE, however, uses sample weights to decorrelate the admissible information from the protected information. This ensures that main task predictions and protected information are conditionally independent given the allowable information to discriminate against.

Applied to a classification task using the Amazon Product Review Dataset, this method results in a predictive model that does not lose much accuracy. Some fairness measures, however, have not reduced significantly. This is explained as a negative and unavoidable side-effect of the fairness constraint that equalizes the predictive rates across the gender groups and imbalanced target classes. But, by assessing the level of discrimination per product category in more detail, the fairness constraint is nearly satisfied. Even though there are limitations and open questions that need to be addressed in future studies to validate the robustness of the method, the framework seems to be effective in improving fairness in the model.

Acknowledgements

I would like to thank my supervisor Professor Bas Donkers at the Erasmus University Rotterdam for allowing me to be involved in this research project. He mentored, motivated and encouraged me throughout the ups and downs during the process. Moreover, Professor Bas Donkers kept challenging me to exceed my own expectations which made me more enthusiastic to pursue a career in research. Thanks to his support and freedom, I had all the resources I needed to conduct research in fairness and explore new deep learning concepts. I also cannot express enough thanks to my family and friends. During the global pandemic, they supported me to keep pushing and take some time off to relax when it was needed. Especially, in these times, it is good to stay curious and work on projects you are passionate about. But, the importance of mental health must not be underestimated.

Contents

1 Introduction

1.1 Machine Learning in Society and Business

Machine Learning (ML) has become an integral part of society, impacting the media we consume, the stories we read, the people we connect to, the places we visit, and the ads we see on the Web. Also, organisations are becoming more reliant on ML since their databases are increasing in size and complexity. Most businesses are mining huge amounts of data about their customers while governments are gathering data on the public for their policy-making decisions. Among different ML models, Deep Learning (DL) models have increased in industry usage. Their popularity can partially be attributed to their state-of-the-art performance in many domains. Although powerful, DL systems are often seen as a 'black box' which does not provide clear insights into how decisions, i.e. predictions, are made on the data. This is problematic and makes it challenging to analyse whether decisions are made based on justified, fair reasons. It is therefore of societal and ethical importance to ask whether algorithmic models might perpetuate existing biases towards particular groups of people sharing sensitive, or so-called protected, attributes such as race and gender. Indeed, there is a growing concern that the use of DL and ML in general can lead, even in the absence of intent, to a lack of fairness. This concern seems to be valid since harmful biases in society could be reflected and accentuated by such models.

What makes it possible for ML models to capture unintended social biases? One option is that ML algorithms are trained on historical data for which it can reflect patterns from the past. Consider algorithmic models that are used for determining credit rating, helping in hiring decisions, or helping judges to make bail decisions for criminal defendants. When a company has hired mostly males for engineering positions or individuals from a demographic minority group continued their criminal activity after being charged with a crime, the ML model would reflect this pattern. It can label female applicants as less suitable for the same engineering position or assign a higher risk of future criminal activity to defendants from the demographic minority group while they are actually innocent (Chen et al., 2018; Courtland, 2018). These discriminatory effects result in unfair and unethical outcomes.

Sensitive attributes do not have to be explicitly available. ML advancements have enabled algorithmic models to learn complex relationships among the data. Therefore, stereotypical effects can be learned by deriving latent sensitive attributes from

the database. For instance, in their quest to show relevant ads to users, digital advertising platforms show a social bias in the delivery of employment and housing ads related to some demographic groups; even when advertisers cannot add demographic information in the settings for their campaign (Sapiezynski et al., 2019). This can occur due to the ad content, such as text, from which the ML algorithm could derive sensitive attributes (Ali et al., 2018). Moreover, Recommender Systems (RS), can also suffer from lack of fairness (Mansoury et al., 2019; Edizel et al., 2019). To resolve rating sparsity and cold-start problem, review-based RS are developed to incorporate textual information into the user modeling and recommending process (Jakob et al., 2009; Zheng, Noroozi, and Yu, 2017; Seo et al., 2017). But, this could expose the system to stereotypical effects which lead to antisocial recommendations.

To mitigate the unfair biases, the source of the sensitive information must be identified in the ML development lifecycle. This lifecycle can be categorized into three stages: preprocessing, inprocessing or so-called model building and postprocessing. Each stage requires a different approach to avoid antisocial predictions. For instance, the protected attribute can be decorrelated with other features in the learning algorithm instead of omitting the sensitive features from the database in the preprocessing stage; which does not remove the bias entirely. The bias could still be present in a latent manner. In some cases, however, it might be acceptable to discriminate among protected groups through the dependency on admissible information. For instance, in the context of RS, products must be recommended based on interest similarity and without discriminating among gender. But, when product interests are dissimilar, it can be viewed as admissible to recommend different products to each gender group. More specifically, it can be reasoned that gender groups that are interested in products from similar categories should be treated similarly; whereas discrimination between the groups with dissimilar interest is considered to be reasonably fair. Thus, the admissible discrimination could ensure that review-based RS are reasonably fair in the predictive outcomes while maintaining as much information as possible.

Even though it requires policymakers, scientists and practitioners to consider complicated ethical questions, it is important to understand the variety of situations in which antisocial biases can manifest, what socially acceptable reasons are to vary among protected groups, and how ethical systems can be designed. Therefore, it is essential to design a robust debiasing framework such that unintended biases can be controlled and mitigated. This thesis proposes an adversarial debiasing framework based on DL methods and focuses on unintended gender biases in product rating predictions. Based on customer reviews and structured data, the framework aims to maintain fairness in the predicted product rating across gender groups while allowing acceptable discrimination through admissible information.

1.2 Research Question and Overview

In Section 2, the basic concepts of DL are addressed, the term 'Fairness' and the Fairness constraints are defined. This to ensure that the reader is familiar with the terminology used in this work and understands the basic mechanisms behind neural networks. Moreover, this Section describes the possible sources of biases from the perspective of the ML development lifecycle.

In the remainder of this work, the adversarial debiasing framework is discussed and validated based on an experiment. There are three sets of research questions that are addressed throughout this paper. The research questions are formulated as follows:

Research Question 1

What adversarial debiasing approaches do exist and are effective to control and mitigate unintended biases in deep learning models?

Section 3 discusses two adversarial methods from recent developments in DL. The first method focuses on mitigating the protected information from hidden representations in the model. The second method, on the other hand, addresses the adversarial setup from the generated predictions to improve fairness. However, both methods are capable of making the model oblivious from the protected information.

Research Questions 2

How can the adversarial learning method be adjusted to debias hidden representations while allowing acceptable discrimination through admissible features?

Section 4 addresses the proposed debiasing method, an adversarial ensemble-based approach to control and mitigate unintended biases. To allow the predictions to discriminate across protected groups only through admissible information, an ensemble of expert model components must be designed. More specifically, each expert model takes different input features such that the adversarial debiasing procedure can be applied in a segregated manner. First, the extended adversarial learning algorithm is discussed and theoretically proved. Afterwards, an extended backpropagation weight update-rule is suggested to improve the effectiveness of the framework.

Research Question 3

To what extent can the adversarial ensemble-based method control and mitigate unintended gender biases in product reviews such that rating predictions and gender information are conditional independent given the product categories?

Section 5 discusses the experiment, model architecture used to implement the adversarial debiasing method and results.[1] First, the database that is used for the experiment is addressed and explored. Afterwards, the created model architecture and evaluation metrics are addressed in more detail. Then, the experiment is performed. The adversarial ensemble-based model is compared against a biased baseline model for which an extensive analysis of the results is conducted. Finally, in Section 6, the thesis is concluded and suggestions for future research is presented.

[1] The code is available in: https://github.com/nvrsmeele/adversarial_debias

2 Background

This thesis starts with an introduction into Deep Learning (DL) and Fairness concepts to ensure familiarity with the terminologies and mechanisms used in this work. Section 2.1 introduces the basic concepts of neural networks. Section 2.2 defines the term Fairness and the different perspectives of the term. Section 2.3 addresses the measurements used to assess whether algorithmic models are generating fair predictions. Finally, Section 2.4 discusses the possible sources of antisocial biases from the perspective of the ML development lifecycle.

2.1 Basic Concepts of Neural Networks

The objective of this thesis is to propose a debiasing framework based on DL methods to control and mitigate unintended biases. But, what is DL? In essence, a DL system trains itself to process and learn from data through an *artificial neural network*. The artificial neural network (NN) resembles the functioning of the human brain which is made up of interconnected neurons. This ensures that the network can learn and make decisions in a more human-like manner (Russell and Norvig, 2009). Like the human brain, a NN is a composed network of interconnected nodes, so-called *perceptrons*. According to (Rosenblatt, 1958), perceptrons are computational units that map some set of input values m to an output value as follows:

$$\phi(\theta_0 + \sum_{i=1}^{m} \theta_i x_i) \tag{2.1}$$

Where x_i is i-th input value, ϕ is the activation function that transforms the input into the output representation, θ_0 is a constant value or so-called bias term and θ_i is the associated link with its weight between connected perceptrons. More specifically, the perceptron computes a weighted sum of the input representation as $\sum_{i=1}^{m} \theta_i x_i$, adds a constant bias term to provide more flexibility and better generalization into the node, and transforms the weighted sum into the output representation by using a (non-)linear activation function (Russell and Norvig, 2009). The node's output representation, however, can be used to feed into a subsequent perceptron for further transformations. It can also be considered as the definite output representation. In this context, the weights Θ are the only trainable variables for each node.

Having defined the computational model for perceptrons, the concept can be extended to form an interconnected network of such nodes. In the most simplistic form, a NN can have three layers of nodes: (i) *input layer* that takes the raw input values and does not apply any computations, (ii) *hidden layer* to transform the values taken from the input layer into the output values or so-called *activation values*, and (iii) *output layer* that computes the actual output representation based on the activation values. This simplistic network is a so-called *feed-forward network* (Goodfellow, Bengio, and Courville, 2016). It connects layers and nodes only in one direction where each node receives input from preceding nodes and generates output values for subsequent nodes. It is important to note that more complex architectures exist where *recurrent networks* are discussed in Section 4. For now, the objective is to ensure familiarity with the most basic concepts of the method.

Consider again the simplistic feed-forward network. Each layer can have a different number of nodes. The input layer, in most cases, has the same number of nodes as the number of input values; where each *input node* takes one input value (Russell and Norvig, 2009). The hidden layer uses the perceptron's computational unit and may vary in the number of nodes. These are often referred to as *hidden nodes*. In practice, the number of hidden nodes used in this layer are seen as a tuneable *hyperparameter*. More specifically, increasing or decreasing the number of hidden nodes affects the network's flexibility where the most optimal number of nodes can be determined by trial-and-error to obtain the best, generalizable performance (Russell and Norvig, 2009). The number of nodes in the output layer, however, depends on the type of problem at hand. For classification problems, this layer can have as many nodes as the number of target classes. This is also known as multi-class classification where it uses a softmax activation function to compute the predicted probabilities per class. When the network is used to solve a binary classification task, the layer has one node with a sigmoidal activation function. For regression tasks, the output layer only needs one node and uses a linear activation function to generate predictive outcomes (Hastie, Tibshirani, and Friedman, 2017). Thus, a single-layer network with only an input and output layer can be used to represent linearly separable functions. A multi-layered network or so-call *deep networks*, on the other hand, with one or more hidden layers are more complex and can be used to map non-linear, convex regions for high-dimensional representations (Russell and Norvig, 2009).

The power behind NN is that it can learn complex, non-linear representations in the given data which, in some cases, outperforms traditional machine learning methods. But, how can these powerful networks be trained? To train a network, the set of weights Θ are updated continuously such that a *loss function*, or so-called *cost function*, is minimized. Thus, the differences between the ground-truth and predicted target values must be small. The general notation of a loss function can be given as:

$$L(\Theta) = \frac{1}{N}\sum_{i=1}^{N} l(f(x_i, \Theta), y_i) \tag{2.2}$$

Where $f(x_i, \Theta)$ is the NNs computational notation to transform the input representation vector x_i for i-th data observation into the output value based on the set of weights Θ in the network (Goodfellow, Bengio, and Courville, 2016). Moreover, y_i is the ground-truth target value and l represents the loss function. Thus, the overall loss $L(\Theta)$ is derived by taking the average over the combined loss values for each i-th example. There are different types of loss functions where the selection depends on the type of problem at hand. For instance, when having a classification tasks, the cross-entropy function is generally used. For regression tasks, the squared error loss function is usually used in practice (Goodfellow, Bengio, and Courville, 2016).

To obtain the optimal set of weights Θ, an *optimization method* is used which utilizes the *backpropagation* algorithm (Russell and Norvig, 2009). Algorithm 1 describes the backpropagation concept according to (Rumelhart, Hinton, and Williams, 1986). At first, the weights are randomly initialized after which the training data is repeatedly passed into the network to obtain the predicted target values. This is known as *forward propagation* or *forward pass*. Then, the loss value is derived for each observation such that the cost can be propagated backwards to compute the weight update direction. The updated weights are determined by taking the gradients of the loss with respect to the weights in the output layer. By iterating backwards through the network, the gradients for every weight is derived from the latter layer via the chain rule. This procedure is repeated until the entire training data went through the system which is referred to as a *training step* or *epoch*. The training step is also repeated multiple times until the model achieves the most optimal prediction performance, i.e. converges at some optimum point, or some other stopping criterion is satisfied.

Algorithm 1: Backpropagation Pseudocode

Initialize network weights (W) with small random values;
for *each training example* x_i **do**
    $prediction_i = Forward\ Pass(x_i, W)$;
    $error_i = Loss(prediction_i, true\ label_i)$;
    for *each layer* (j) **do**
        compute all $\frac{\partial error_i}{\partial W_j}$;
        update network weights $W \leftarrow W - \eta \frac{\partial error_i}{\partial W_j}$;
    end
end

Algorithm 1 shows the backpropagation algorithm which updates weights per training example. This is known as *stochastic gradient descent (SGD)*. However, this approach takes time to achieve one training step for large datasets and the loss decreases with fluctuations overtime (Goodfellow, Bengio, and Courville, 2016). Another approach that can be used is *batch gradient descent (BGD)* which takes all training data into consideration to achieve one training step (Goodfellow, Bengio, and Courville, 2016). This considers the average gradient of all training observations and uses the mean gradient to update the weights. The advantage of BGD is that it smoothens the loss over the epochs and moves somewhat directly towards an optimum solution. A disadvantage is that it requires more computational power compared to SGD since all training examples must pass through the system before the weights are updated (Goodfellow, Bengio, and Courville, 2016). To leverage both approaches and their advantages, *mini-batch gradient descent (MGD)* can be used. This uses a batch of fixed number of training observations which is less than the entire training database. Afterwards, the algorithm runs each mini-batch through the system and updates the weights accordingly (Goodfellow, Bengio, and Courville, 2016).

Moreover, the backpropagation method uses some optimization method to update weights. There are multiple types of techniques that can be used to effectively update weights. However, the general formula for the weight update-rule is:

$$\Theta_t \leftarrow \Theta_{t-1} - \eta \frac{\partial L(\Theta_{t-1})}{\partial \Theta_{t-1}} \tag{2.3}$$

Where Θ_t are the newly updated weights at training step t, Θ_{t-1} are the weights from the previous training step that are being updated, η is the learning rate that determines the step size at each weight update iteration and $\frac{\partial L(\Theta_{t-1})}{\partial \Theta_{t-1}}$ is the gradient of the loss with respect to the weights (Goodfellow, Bengio, and Courville, 2016). The learning rate is often seen as a tuneable hyperparameter since it can affect the learning speed as well as overall performance of the network. There are two categories of optimization methods: (i) methods with fixed learning rate such as *stochastic gradient descent* and *(mini-)batch gradient descent* (Goodfellow, Bengio, and Courville, 2016), and (ii) methods with adaptive learning rate schedules such as *Adam* (Kingma and Ba, 2015) and *RMSProp* (Hinton, 2012). Instead of using the fixed learning rate, the adaptive learning rate methods adjust the learning rate throughout the training procedure in response to the current performance of the model at each training step. These methods utilizes (mini-)batch gradient descent approaches to improve their efficiency. Some studies have shown that the *Adam* method avoids local minima and saddle points, and converges fast without the requirement to tune many parameters compared to other methods (Kingma and Ba, 2015; Ruder, 2016). This is achieved due to the weight update-rule that uses adaptive learning rates. More specifically, this method adjusts the learning rate at each training step by considering the results of the weight updates in the preceding training step (Kingma and Ba, 2015).

This thesis proposes a debiasing framework based on NNs. More specifically, it utilizes deep, interconnected networks for which it is essential to be familiar with the basic concepts and terminology used in this Section. Moreover, it is important to note that this work focuses on fairness in classification tasks and uses the terms *neural networks* and *deep learning* interchangeably without making any distinction between the simplistic perceptron network or a deep, multi-layered model.

2.2 Defining Fairness

Most traditional Machine Learning (ML) models are trained to process and learn from data in a transparent manner. DL, however, trains itself given the boundaries via the network's architecture. It transforms the input representation via computational units into an output representation as discussed in Section 2.1. These models are, therefore, not directly transparent in how predicted decisions are made for which it is referred to as a 'black box'. This can be problematic and challenging. As these systems often impact groups of individuals and society, it could generate unfair outcomes in a non-transparent manner. To determine what *fairness* and *discrimination* is, it becomes even a more difficult challenge. These concepts are already complicated since it requires critical questions about social and ethical aspects. A good starting point is to connect discrimination in ML to existing legal frameworks. Many countries have anti-discrimination laws to avoid unfair treatments based on on so-called *protected* or *sensitive attributes* such as age, gender and race (Zafar et al., 2017). In general, these regulations evaluate fairness based on two notions:

Disparate Treatment These are decisions that are explicitly based on sensitive information about individuals (Zafar et al., 2017). DL systems, however, are not designed and trained to explicitly discriminate on the basis of the protected attributes. Still, it can produce predictions based on these sensitive information. Section 2.1 have shown that models are trained by minimizing some loss function which is not constrained to act fair. Thus, the optimized cost functions can enable models to generate unfair predictive outcomes based on the reflection of antisocial patterns in the data.

Disparate Impact These are decisions that unintentionally discriminate against protected individuals (Zafar et al., 2017). More specifically, if sensitive features are not explicitly included in the database, the protected information can be derived from nonsensitive features that may be correlated with the sensitive features (Soundarajan and Clausen, 2018). Thus, DL systems can produce discriminatory outcomes even without the intent to act unfair. This can be referred to as *unintended bias*.

In the optimal situation, it is desired to have an algorithmic model that does not involve disparate treatment and disparate impact. However, it is challenging to simultaneously control for both notions of unfairness. The naïve solution to avoid disparate treatment is to ensure that the model cannot use the protected attribute. In practice, though, the protected attributes are often not available due to privacy

and other regulations (Sapiezynski et al., 2019; Mansoury et al., 2019; Edizel et al., 2019). This does not necessarily imply that both notions are controlled. As addressed earlier, the nonsensitive and sensitive features may be correlated which enables the system to violate the notion of disparate impact (Chen et al., 2019; Chouldechova, 2017). This makes the definition and assessment of fairness more complex.

Most studies on fairness in ML have attempted to simplify the evaluation procedure by focussing on classification tasks (Dwork et al., 2012; Binns, 2020; Chouldechova and Roth, 2018). As mentioned, the main challenge for DL systems is to obtain a well-optimized loss function while avoiding any discriminatory outcome against protected groups. But, what are the levels on which fairness can be measured? There are two levels to evaluate fairness:

Group Fairness This addresses fairness from the perspective of generalized groups such as racial, gender or age groups. More formally, group fairness is the property that demographic groups have equalized outcomes across protected and non-protected groups (Dwork et al., 2012). In statistical terms, however, group fairness can also be referred to as *Demographic Parity* or *statistical parity*. To ensure that group fairness is maintained, the average correctly predicted outcomes must be equal for each group. Section 2.3 formalizes the statistical measures in more detail.

Individual Fairness This notion involves constraints on individualistic level rather than on an averaged group quantity level (Chouldechova and Roth, 2018). (Dwork et al., 2012) argues that "similar individuals should be treated similarly". Therefore, to satisfy this notion, a distance measure is often used to determine how similar individuals are based on features related to the prediction task. More specifically, the individuals are mapped into a feature space without including the protect feature after which the distance is measured among the individuals in database. Then, individuals that have the smallest distance between them are clustered in the same group. When having new individuals for which a prediction must be generated, the distance among all clusters and the new individuals are derived after which they are assigned to the group with the smallest distance value. On this basis, individuals in the same group are treated more similarly without considering the protected information. In some cases, individual fairness can be seen fairer compared to group fairness. Indeed, group fairness ensures that protected and non-protected groups have equalized outcomes. This could, for example, result in unfair predictions to those individuals who are more 'qualified' than the ones who received positive outcomes to equalize the average correctly prediction rates.

The fairness notions are quite complex and require the consideration of complicated ethical questions. Since individual fairness require some assumptions about the selected similarity metric and set of features used to cluster individuals, this thesis focuses on *group fairness*. This notion is used to control and mitigate unintended gender bias. The term *bias* has many definitions in ML. In this work, bias is defined

as a type that introduces discriminatory patterns in algorithmic models. It is also important to note that within group fairness more complicated social relationships can exist such as *subgroup fairness* or so-called *fairness gerrymandering*. This suggests that a model can act fair on each individual group while it violates constraints on subgroup level (Kearns et al., 2018). For simplicity reasons, however, this thesis considers group fairness and does not analyse the effects on subgroup level.

Example: Why could Statistical Parity be insufficient?

Let *gender* be the protected attribute where the objective is to not discriminate among *male* and *female*. Suppose the University of Fairness uses an algorithmic model to help them decide whom to admit to their MBA-program. The admission decision is mainly based on GMAT scores with the acceptance threshold being a score of 600. When the model was created, the training database contained more male than female applicants as shown in Table 2.1. Moreover, the male applicants tend to be wealthier for which they can afford GMAT preparation courses and take multiple tests. The female applicants, on the other hand, take the test once with less practice and preparation. This results on average in higher GMAT scores for male applicants. But, it turns out that both groups have equal chances to succeed in the program.

TABLE 2.1: MBA-applicants at the University of Fairness

Applicant	GMAT Score	Succeeds
Male	720	Yes
Male	695	Yes
Male	675	Yes
Male	635	No
Male	610	Yes
Male	600	Yes
Male	595	No
Female	575	Yes
Male	560	Yes
Female	545	Yes
Female	520	No
Female	500	No

By using the acceptance threshold of a 600-score, the prediction model makes exactly three mistakes: one male applicant above the cutoff and one male and two female applicants below the cutoff. This means that two successful female applicants are falsely rejected whereas only one successful male

applicant is falsely rejected. In this case, female applicants can complain and sue the university for being sexists. To avoid such an altercation and justify for the discrimination, the university's executive board could decide between two options: (i) increase the cutoff to 610 which results in another falsely rejected male applicant, or (ii) decrease the GMAT score cutoff to 575 which correctly accepts a female applicant. Both options equalizes the false rejection rate among gender groups. But, it worsens the model's accuracy as more applicants are being falsely rejected or accepted to the MBA-program.

2.3 Measurements of Demographic Fairness

There are several evaluation criteria that measure different aspects of inequality and fairness. As addressed in Section 2.2, this thesis is focused on group fairness which can be assessed through several metrics. However, the two main measures are: (i) Statistical Parity or so-called *Demographic Parity* (Zemel et al., 2013) and (ii) *Equalized Odds* (Hardt, Price, and Srebro, 2016). To define these metrics, consider the University of Fairness again that uses an algorithmic model to help decide whom to admit to the MBA-program. The objective is to predict whether an applicant will succeed in the program. More formally, the goal is to train a binary predictor $\hat{Y}$ from features x_i based on labeled training data y_i while ensuring that each $\hat{Y}$ is nondiscriminatory with respect to a protected attribute Z. Moreover, $y_i = 0$ implies failure and $y_i = 1$ represents that the applicant will succeed in the MBA-program where the sensitive feature $Z \in (0, 1)$ is either male or female. The fairness measures can be defined as:

Demographic Parity This metric holds when the prediction $\hat{Y}$ is independent of the sensitive feature Z (Hardt, Price, and Srebro, 2016; Zemel et al., 2013). The relationship between the protected feature and predictions should not be correlated. Thus, this constraint requires that the proportion of each protected group should receive equal positive outcomes with respect to a target class. In the university's admission example, this means that the number of predicted successful male and female applicants must be equalized. More formally, the constraint can be denoted as:

$$Pr(\hat{Y} = y|Z = 0) = Pr(\hat{Y} = y|Z = 1), y \in (0, 1) \tag{2.4}$$

Equalized Odds This measure holds when predictions $\hat{Y}$ and protected attribute Z are conditional independent on the target outcome Y (Hardt, Price, and Srebro, 2016). It allows $\hat{Y}$ to depend on Z through the actual outcome Y which is not allowed in the demographic parity constraint. This also considers that each protected group can be distributed differently in terms of the actual outcome Y. More specifically, for outcome $y = 1$, the predictive outcomes $\hat{Y}$ must have equal true positive rates across the protected groups Z where it also equalizes the false positive rates for

$y = 0$ (Hardt, Price, and Srebro, 2016). In the university's admission example, this means that male and female applications have equal chance of being correctly and falsely classified as a candidate that will succeed in the MBA-program. Hence, this constraint can be formalized as:

$$Pr(\hat{Y} = 1|Z = 0, Y = y) = Pr(\hat{Y} = 1|Z = 1, Y = y), y \in (0, 1) \tag{2.5}$$

Even though both measures are used to assess fairness, these constraints are difficult to satisfy simultaneously (Dwork et al., 2012; Hardt, Price, and Srebro, 2016; Kleinberg, Mullainathan, and Raghavan, 2016). Demographic parity can ignore qualified individuals in $Z = 0$ and accept unqualified individuals in $Z = 1$ to equalize the predicted proportion across the protected groups. This could occur when the training data is small or imbalanced in the sensitive attribute. As addressed in Section 2.2, group fairness can be interpreted as unfair based on this constraint. Moreover, the prediction performance can also be affected by this measure since it forces the model to balance the true positive rates. (Hardt, Price, and Srebro, 2016) argued that the performance is affected since demographic parity does not allow the classifier to depend on the actual outcome Y. The equalized odds metric, however, does allow dependency between the prediction and the actual outcome Y. Thus, both measures addresses fairness from different perspectives that conflict with each other when both are attempted to satisfy simultaneously. It should be noted that there exists another measure that relaxes some requirements of the equalized odds metric. This is the *equalized opportunity* constraint that only equalizes the true positive rates among the protected feature given the actual outcomes (Hardt, Price, and Srebro, 2016).

Even though demographic parity and equalized odds can be used, these constraints are focused on maintaining fairness in the overall model. In this thesis, however, the objective is to achieve fairness such that only acceptable discrimination can occur through the dependency on admissible information. Section 1 have addressed that such discrimination can be considered to be reasonably fair. Therefore, the following adjusted fairness constraint is formalized:

$$P(\hat{y}|z = 0, x_{adm}) = P(\hat{y}|z = 1, x_{adm}) \tag{2.6}$$

where $\hat{y}$ represents the predicted class, $z \in (0, 1)$ is the protected feature and x_{adm} are the admissible data. Thus, the constraint holds when predictions $\hat{y}$ and protected feature z are conditionally independent given the admissible features. In the university case, this notion means that both male and female applicants should have equal chance of being accepted or rejected for the MBA-program given, for example, their academic background. More specifically, the number of predicted (un)successful male and female applicants must be equalized. Thus, this fairness notion is used to assess the effectiveness of the proposed debiasing framework.

2.4 Sources of Unintended Demographic Biases

The previous Sections have addressed the background context for this thesis in which the term *fairness* and evaluation metrics are defined. Before the existing and proposed debiasing methods can be introduced and assessed, it is essential to understand how unintended biases can occur in algorithmic models. In other words, what are the potential sources from which unintended biases can emanate? The main sources for distortions can be identified in the ML development lifecycle. The stages in the lifecycle are: (i) data collection, (ii) data preprocessing, (iii) model building and (iv) model deployment and inference. However, the three main sources in the development lifecycle that could introduce unintended biases are discussed below.

Dataset The database used to train ML systems can contain human, unintended biases. Based on historical patterns in the training data, the models can reflect and perpetuate the patterns from the past. Of course, this does not mean that all data used for training contain unfair patterns. But, it can be possible that it is skewed or so-called *imbalanced* towards some protected group. Since algorithmic systems are optimized to fit the data, it can naturally reflect these imbalanced patterns which results in unfair predictive outcomes (Chouldechova and Roth, 2018). There are studies that have identified such biases (Chen et al., 2018; Angwin et al., 2016). For instance, (Angwin et al., 2016) found that a prediction system used to help judges to make bail decisions for criminal defendants was biased. The system predicted high recidivism rates that were correlated with darker skin color. This antisocial prediction behavior was caused by the skewed database used to train the model in which dark skinned individuals were overrepresented over light skinned individuals. In practice, *preprocessing methods* can be used to control and mitigate these data imbalances. Some studies have shown the effectiveness of data augmentation and reweighting methods (Dixon et al., 2018; Kamiran and Calders, 2011). Moreover, *inprocessing methods* can also be used to constrain the learning algorithm based on optimization approaches (Zemel et al., 2013). It is important to note that each method could have unintended side-effects for which the approach selection must be done carefully. For instance, when constraining the learning algorithm, the overall prediction accuracy may be affected. Or, when sampling methods are used, the training database can change significantly in size and could lose valuable information.

Learning algorithm This enables a system to produce antisocial outcomes without the intent to discriminate. As addressed in Section 2.2, ML systems are trained to minimize some optimization function which is not constrained to achieve fairness. The models can, therefore, sacrifice fairness to achieve higher prediction accuracies. Though there are some *inprocessing methods* that can be used to mitigate these unintended biases. One option is to add fairness constraints to the loss function via regularization terms (Ross, Hughes, and Doshi-Velez, 2017; Liu and Avci, 2019; Du et al., 2019). This minimizes the association between the predictive outcomes and sensitive

attributes where it forces the system to focus more on correcting features relevant to the task. However, regularization requires feature-wise annotations that specifies whether each input feature correlates with the protected attribute (Ross, Hughes, and Doshi-Velez, 2017; Du et al., 2019). Moreover, the learning algorithm could also learn the sensitive relationships through *encoded data representations* when the protected features is not explicitly available in the database (Chen et al., 2019). This is well-studied in the field of natural language processing where embeddings are used to create text representations. Some work has found that these text representations can contain biases (Bolukbasi et al., 2016; Caliskan, Bryson, and Narayanan, 2017; Zhao et al., 2019; Kurita et al., 2019). Usually, mitigating biases via the learning algorithm is an effective approach but it is often complex to identify the actual source of the bias. This thesis is focused on the unintended biases via the learning algorithm.

Decision-level The algorithmic decisions made on calibrated thresholds can introduce unintended biases in the model deployment and inference stage or so-called *postprocessing stage* of the ML development lifecycle. More specifically, postprocessing calibration considers the system's predictions and sensitive attributes to adjust the generated outcomes at inference time (Zhao et al., 2017; Hardt, Price, and Srebro, 2016). It can, therefore, be used to enforce the model's performance to satisfy some specific fairness measure which is proved to be effective to reduce discriminatory outcomes (Kleinberg, Mullainathan, and Raghavan, 2016; Chouldechova, 2017). (Hardt, Price, and Srebro, 2016) shows how a probabilistic classifier can calibrate demographic-wise thresholds to achieve some fairness constraints. However, when only postprocessing methods are used to mitigate unintended bias, it can be complex to ensure model fairness completely. As discussed in Section 2.3, the main fairness measures cannot be satisfied simultaneously.

Since the fairness objective for this thesis is to allow dependency between the predictions and protected information only through admissible data, this work is focused on inprocessing methods to address unintended group biases. More specifically, this thesis proposes an adversarial debiasing framework to control and mitigate unintended biases. This method is validated in the setting of omitting unintended gender biases in product rating predictions. Based on customer reviews and structured data, predictions are generated where biases are removed from the encoded representations while allowing acceptable differences through the product categories. By considering the adjusted fairness constraint as addressed in Section 2.3, the framework is validated and compared to some baseline model.

3 Understanding Adversarial Representation Learning Methods

This Section introduces existing adversarial learning methods that are able to mitigate unintended biases and ensure fairness in models. First, an adversarial representation learning method is introduced in Section 3.1. Afterwards, the model specifications are given and learning algorithm is explained. Then, in Section 3.2, an adversarial method that uses predictions instead of representations is discussed. This utilizes another model structure to ensure that fairness constraints are satisfied.

3.1 Adversarial Invariant Feature Learning

This thesis is interested in identifying adversarial debiasing approaches that can control and mitigate unintended biases in Deep Learning (DL) classification models. As discussed in Section 2.4, antisocial behaviour in algorithmic systems can be embedded through several sources. One source is the learning algorithm used to train models. Since data representations can contain biases, the protected attributes does not have to be explicitly available in databases. Advanced learning algorithms can enable models to reflect the implicit biases in predictive outcomes. This can have a severe impact on the fairness maintained in the model without the intent to be unfair. For instance, when text data is used for Natural Language Processing (NLP) tasks, the antisocial biases can implicitly be present in the encoded representations which can be captured by algorithmic models. Many researchers have identified gender biases in existing language modeling techniques such as Word2Vec (Mikolov et al., 2013; Bolukbasi et al., 2016), GloVe (Caliskan, Bryson, and Narayanan, 2017; Prost, Thain, and Bolukbasi, 2019), ELMo (Zhao et al., 2019) and, even in the state-of-the-art method, BERT (Kurita et al., 2019). Other studies also examined the impact of these biases in sentiment analysis tasks (Kiritchenko and Mohammad, 2018). Thus, when the protected features are not explicitly available, this does not imply that fairness is maintained in DL models. The systems could still derive the sensitive attributes and learn the discriminatory patterns in hidden representations through the algorithm. One method that can be used to mitigate such unintended biases is *adversarial learning*. This Section discusses an adversarial approach suggested by (Ganin and Lempitsky, 2015; Beutel et al., 2017; Xie et al., 2017; Elazar and Goldberg, 2018) that focuses on debiasing the hidden representation of the model.

3.1.1 Specifications of the Learning Setup

The adversarial learning method is used to hide information from the main prediction model by penalizing the overall loss function. In essence, this method simultaneously trains two networks: (i) the main task model or so-called *predictor* is learned to be maximally informative for the main task and (ii) the adversarial model or so-called *adversary* is trained such that its ability to predict the protected attribute based on the predictor's encoded representations is minimized. (Elazar and Goldberg, 2018) have experimented with this method. Their study was focused on debiasing downstream models via adversarial learning such that the predictive power for some demographic features, i.e. race, gender and age, in raw text is reduced. Figure 3.1 shows the learning setup used in their work which follows earlier studies from (Ganin and Lempitsky, 2015; Beutel et al., 2017; Xie et al., 2017).

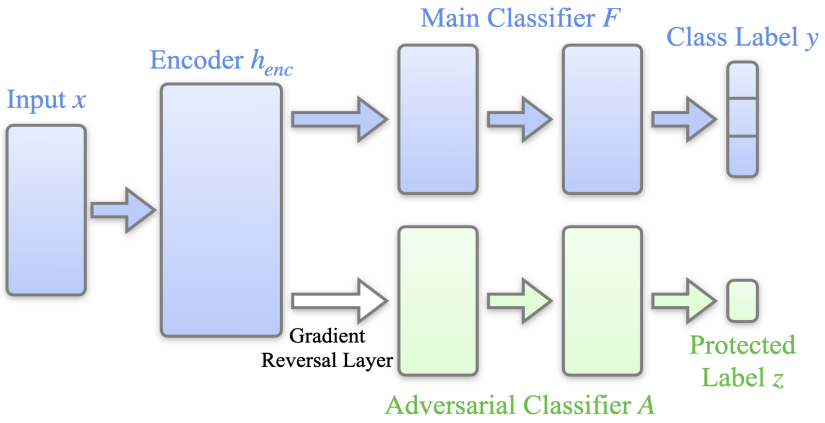

FIGURE 3.1: Model learning setup by (Elazar and Goldberg, 2018)

To ensure fairness and invariant representations, the learning setup is inspired by recent advancements in adversarial learning (Goodfellow et al., 2014) and can be referred to as a minimax game between three players: (i) an *encoder* which transforms the observed data into a feature space, (ii) a *predictor* which uses the representation to generate main task predictions, and (iii) an *adversary* which uses the encoder's representation to identify the protected information that must be mitigated from the features. Moreover, (Elazar and Goldberg, 2018) have used a fourth model or so-called *attacker* model that uses the encoder's representation to verify whether it is invariant for the protected feature after the adversarial minimax game is played. More specifically, consider a database that contains text documents $x_1, ..., x_n$ with a protected attribute z and task labels $y_1, ..., y_n$. The objective is to train a predictor F that accurately predicts the task labels y while ensuring that the encoder's representation is invariant for the protected feature z. To achieve this objective, the encoder transforms x into representation h_{enc}. Based on representation h_{enc}, the predictor F is trained to model the conditional distribution $p(y|h_{enc})$. However, the F is considered to be fair when h_{enc} does not depend on z or if h_{enc} cannot be used to

predict z. To make h_{enc} invariant for z, the adversary A is simultaneously trained to fit $p(z|h_{enc})$ where h_{enc} attempts to make A fail in its prediction. Thus, while F and A are optimized to minimize their loss function, the encoder h_{enc} is trained such that it is maximally informative for F and minimizes A's ability to predict z. To assess whether the main task predictions are invariant for z after training, (Elazar and Goldberg, 2018) have used an attacker model Att that attempts to predict z based on h_{enc}. The difference between A and Att, however, is that A is considered in the adversarial game during training whereas Att is separately trained by using the pre-trained h_{enc}. When Att is not able to predict z above chance or $p(z) > 0.5$, the model is considered to be fair in generating its predictions.

3.1.2 Adversarial Minimax Algorithm

The adversarial learning algorithm between the encoder h_{enc}, predictor F and adversary A focuses on ensuring that h_{enc} is invariant for the protected information while it remain maximally informative for F. In this setting, the predictor F is trained to fit the distribution $q_f(y|h_{enc})$ whereas the adversary A attempts to model $q_f(z|h_{enc})$. More specifically, (Ganin and Lempitsky, 2015; Xie et al., 2017) have suggested an adversarial minimax game that jointly optimizes the following:

$$\min_{h_{enc},F} \max_{A} J(h_{enc}, F, A) \tag{3.1}$$

$$J(h_{enc}, F, A) = \mathop{\mathbb{E}}_{h_{enc},z,y\sim p(h_{enc},z,y)} [-\log q_f(y|h_{enc}) + \gamma \log q_a(z|h_{enc})] \tag{3.2}$$

Where γ controls the invariance trade-off between the two objective functions for F and A, $\log q_a(z|h_{enc})$ and $\log q_f(y|h_{enc})$ are the conditional entropies and $p(h_{enc}, z, y)$ is the joint distribution. The conditional entropies, however, have opposite signs with $\log q_a(z|h_{enc})$ and and $-\log q_f(y|h_{enc})$. This ensures that the adversarial game makes h_{enc} oblivious for the sensitive information while maintaining most of the predictive information for F. More specifically, the entropy metric measures the prediction certainty and is derived by taking $-\log_2(P)$ with P as the probability for a specific event (Goodfellow, Bengio, and Courville, 2016). Since P ranges between $[0, 1]$, the logarithm obtains negative values in the range between $[-\infty, 0]$ for which the negative logarithm it taken to derive positive values that ranges between $[0, \infty]$. Thus, when the metric obtains an entropy value near zero, the model is considered to be accurate in generating the correct predictions. On the other hand, when the measure approaches ∞, the predictions generated by the system are highly uncertain to be correct (Goodfellow, Bengio, and Courville, 2016). Hence, by considering $\log q_a(z|h_{enc})$ in equation (3.2), the entropy values in A are negative and penalizes F's entropy metric when h_{enc} is predictive for z.

In both (Xie et al., 2017) and (Elazar and Goldberg, 2018), the adversarial learning algorithm is implemented by using the Gradient-Reversal Layer (GRL) method as proposed by (Ganin and Lempitsky, 2015). This layer is inserted between h_{enc} and A, and acts as an identity transform during the forward pass. Then, the gradients of A's loss with respect to the trainable weights in h_{enc} and A are transformed by $-\lambda$ during backpropagation. Here, $-\lambda$ simply multiplies the gradients with -1 such that the gradients are backpropagated with opposite signs. In this case, the encoder receives the gradients with opposite sign from the adversary multiplied by the invariance parameter γ. Thus, by jointly optimizing (3.2) and using the GRL method, representation h_{enc} becomes maximally informative for the predictor while it minimizes the adversary's ability to predict the protected feature (Xie et al., 2017; Elazar and Goldberg, 2018). Therefore, the learning algorithm is referred to as an adversarial minimiax game. Even though the adversarial method should ensure that the hidden representations h_{enc} are invariant for the protected information, (Elazar and Goldberg, 2018) have shown that a fair amount of the protected information can still remain in the model. However, it is doubtful that this consistently occurs in different settings since the adversary aims to remove all sensitive information from the encoded representation in the system. A possible reason that could explain their finding is that the adversary have not been stabilized during training, i.e. converged, for which the learning algorithm has to be trained over more training steps. When the model does not converge, the hidden representations can still contain some sensitive information which can be reflected in the prediction task at hand.

3.2 Adversarial Debiasing Through Predictions

The second adversarial method that can be used to effectively mitigate protected information and ensure fairness in algorithmic models addresses the problem from another perspective. Where the adversarial game in Section 3.1 was inspired by recent advancements in adversarial learning, (Zhang, Lemoine, and Mitchell, 2018) have presented a framework that follows the Generative Adversarial Network (GAN) setup from (Goodfellow et al., 2014). More specifically, the presented framework ensures that the overall model is oblivious from the protected attribute by considering the predictive outcomes as starting point in the adversarial game. This Section discusses the adversarial setup and learning algorithm proposed by (Zhang, Lemoine, and Mitchell, 2018) in more detail.

3.2.1 Specifications of the Learning Setup

As briefly mentioned, the adversarial learning method proposed by (Zhang, Lemoine, and Mitchell, 2018) replicates the GAN setup as suggested by (Goodfellow et al., 2014) in which both a main classifier or so-called predictor F and adversary A are trained simultaneously. The adversarial game, however, separates the models such that it only considers these two components in the algorithm. More specifically,

predictor F generates the main predictions where adversary A is trained to predict the protected attribute based on the predicted values from F. Thus, F is trained to generate main predictions as accurately as possible while attempting to fool A such that A's ability to predict the sensitive feature is minimized. This is different from the learning setup used by (Ganin and Lempitsky, 2015; Xie et al., 2017; Elazar and Goldberg, 2018; Beutel et al., 2017). In their work, the adversarial minimax game was played between three components: (i) an encoder, (ii) predictor and (iii) adversary. This setup was focused on making the encoder's hidden representation invariant of the protected information through the interaction with the adversary. Figure 3.2 shows the model structure used by (Zhang, Lemoine, and Mitchell, 2018).

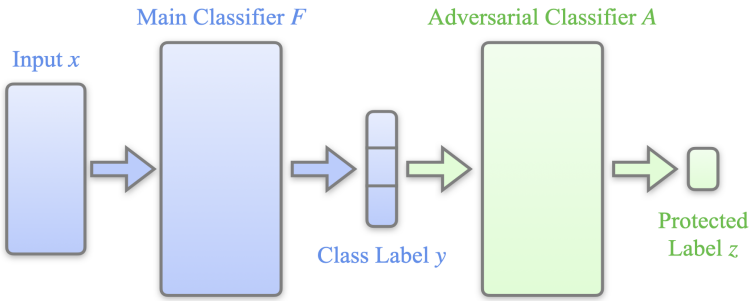

FIGURE 3.2: Model learning setup by (Zhang, Lemoine, and Mitchell, 2018)

To ensure that F is invariant for the protected information, (Zhang, Lemoine, and Mitchell, 2018) have initialized the learning setup such that the predictive outcomes from F are fed into A. Consider a database that contains samples $x_1, ..., x_n$ with a protected attribute z and task labels $y_1, ..., y_n$. In this setting, predictor F must be trained to accurately predict y based on x and independently from the sensitive information z. More specifically, predictor F models the conditional distribution $p(y|x)$ whereas adversary A uses the main predictions $\hat{y}$ to fit $p(z|\hat{y})$. It should be noted that a classification problem is considered for which $\hat{y}$ is referred to the derived softmax or sigmoid values. Thus, by training F to accurately predict y and minimizing A's ability to predict z, the predictor F becomes oblivious for the protected information for which fairness can be maintained in the main predictive outcomes. In other words, the main predictions $\hat{y}$ are decorrelated from the sensitive feature z through the adversarial game between predictor F and adversary A. (Zhang, Lemoine, and Mitchell, 2018) have stated and proved the theoretical guarantees that the demographic parity and equalized odds fairness constraints can be satisfied by using this adversarial setup. Indeed, (Zhang, Lemoine, and Mitchell, 2018) have achieved better results on the equalized odds constraint than (Beutel et al., 2017).

3.2.2 Adjusted Backpropagation Weight Update-rules

In essence, the adversarial algorithm used by (Zhang, Lemoine, and Mitchell, 2018) is comparable to the learning method as suggested by (Ganin and Lempitsky, 2015; Beutel et al., 2017; Xie et al., 2017; Elazar and Goldberg, 2018) and as discussed in Section 3.1.2. A similar objective function is defined which is focused on maintaining fairness in the main predictions generated by predictor F. Consider the following adversarial game that is jointly optimized as:

$$\min_{F} \max_{A} J(F, A) \tag{3.3}$$

$$J(F, A) = \mathop{\mathbb{E}}_{z,y \sim p(z,y)} [-\log q_f(y|x) + \gamma \log q_a(z|\hat{y})] \tag{3.4}$$

Where γ controls the degree of debiasing in predictor F based on adversary's A objective function, $\log q_a(z|\hat{y})$ and $\log q_f(y|x)$ are the conditional entropies, and $p(z, y)$ is the joint distribution. Thus, the same minimax algorithm is used as addressed in Section 3.1.2 in which the gradients are reversed in its sign via the GRL method. However, (Zhang, Lemoine, and Mitchell, 2018) have made an improvement in the weight update-rule for the trainable weights in predictor F. This to ensure that the gradients and updated weights in F cannot help A during the training procedure. More specifically, the adjusted weight update-rule can be formalized as:

$$\theta_f \leftarrow \theta_{f-1} - \eta(\frac{\partial L_f}{\partial \theta_{f-1}} - proj_{\frac{\partial L_a}{\partial \theta_{f-1}}} \frac{\partial L_f}{\partial \theta_{f-1}} - \gamma \frac{\partial L_a}{\partial \theta_{f-1}}) \tag{3.5}$$

$$\theta_a \leftarrow \theta_{a-1} - \eta \frac{\partial L_a}{\partial \theta_{a-1}} \tag{3.6}$$

Where L_f and L_a defines the loss functions for predictor F and adversary A respectively, θ_f and θ_a represent the trainable weight matrices in F and A, η is the learning rate, γ is the invariant trade-off parameter and the updated weights are derived by the partial derivatives of the loss with respect to the trainable weight matrix at each training step. It can be seen that adversary A updates its weights θ_a by following the standard backpropagation method as introduced in Section 2.1. But, when implementing the adversarial game, predictor F uses an adjusted backpropagation setting compared to (Ganin and Lempitsky, 2015; Xie et al., 2017; Beutel et al., 2017; Elazar and Goldberg, 2018). The interaction between F and A still uses the identity transform during the forward and backward pass as suggested by (Ganin and Lempitsky, 2015). However, an additional term is subtracted in (3.5) which is the middle term $proj_{\frac{\partial L_a}{\partial \theta_{f-1}}} \frac{\partial L_f}{\partial \theta_{f-1}}$. This ensures that F's gradients are orthogonal to A's gradients at each training step. In other words, it prevents F's updated weights and its gradients from

moving in a direction that can help A decrease its loss which could result in a lack of convergence to the optimum during training.

Hence, the approach suggested by (Zhang, Lemoine, and Mitchell, 2018) seems more robust to obtain unbiased results than the learning algorithms used by (Elazar and Goldberg, 2018; Xie et al., 2017; Ganin and Lempitsky, 2015; Beutel et al., 2017). It considers the possibility that the predictor and adversary can help each other during training. Even though robust, both adversarial methods are tested and validated when the entire system must be debiased. In other words, the unintended biases cannot be mitigated while controlling stereotypical biases that are acceptable at the same time in these approaches. This can be a disadvantage in practice. As addressed in Section 1, in some cases, acceptable discrimination can be allowed in the predictive outcomes only through admissible information. Based on the existing adversarial methods, the question remains whether an improved debiasing framework can be designed to mitigate unintended biases while controlling stereotypical biases. Section 4 proposes such a framework that extends the adversarial learning algorithm to control acceptable biases through admissible information.

4 Adversarial Ensemble-based Framework

This Section introduces the proposed adversarial ensemble-based framework which is build upon the adversarial algorithms as discussed in Section 3. The objective is to generate predictions without discriminating among protected groups while allowing acceptable dependency between predictive outcomes and sensitive information through admissible features. In Section 4.1, the improved adversarial learning algorithm is theoretically proved. Moreover, it is theoretically proved that a sample weighting scheme is required to ensure that the formulated fairness constraint holds. Afterwards, in Section 4.2, the backpropagation weight update-rule is extended to maintain the effectiveness of the adversarial ensemble-based framework.

4.1 Extending the Adversarial Learning Algorithm

The aim of this thesis is to propose an adversarial ensemble-based framework to control and mitigate unintended gender biases in (un)structured data representations. As addressed in Section 3, there exist two adversarial debiasing approaches that are effective and efficient to use in deep learning (DL) models. However, the learning setups were designed to omit all protected information in the model. To allow acceptable dependency between predictions and protected features, the bias mitigation procedure must be controlled in a segregated manner. This work validates and applies the suggested debiasing framework to a multi-class classification problem. Based on unstructured customer reviews, the model must generate product rating class predictions without discriminating among gender; while allowing justified dependency between class predictions and gender through permissible structured features. Section 5 discusses the collected data, model architecture used in the experiment and experimental results in more detail. For now, it is important to know that the database contains unstructured text attributes referred to as *contaminated* features, and structured features defined as *admissible* features where both include discriminatory stereotypical gender biases. More specifically, the model removes the protected information from contaminated features by leveraging the adversarial minimax game while controlling the dependency between main task predictions and the protected attribute through admissible features. It should be noted that this work considers the contaminated and admissible features to be unstructured

and structured respectively. However, in practice, the features can be substituted or mixed with different data types which depends on the problem at hand. Figure 4.1 shows the overall model structure.

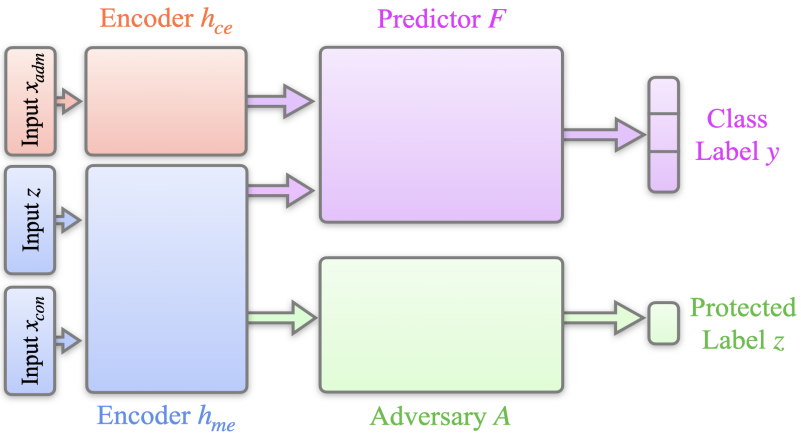

FIGURE 4.1: Adversarial Ensemble-based Model Setup

4.1.1 Specifications of the Learning Setup

The learning setup is closely related to (Elazar and Goldberg, 2018; Ganin and Lempitsky, 2015; Xie et al., 2017). But, as discussed in Section 3.1, their adversarial debiasing method is designed to mitigate all protected information in the model. To allow acceptable discrimination in the predictive outcomes through admissible information, the proposed debiasing framework follows the *Product of Experts* structure as shown in Figure 4.1 and suggested by (Hinton, 2002). Consider the database that contains samples $x \in X$ with structured $x_{s1}, .., x_{si}$, unstructured $x_{u1}, ..., x_{ui}$, and protected attributes z, and ground-truth labels $y_1, ..., y_i$. Moreover, the structured x_s and unstructured x_u features are considered to be the admissible x_{adm} and contaminated x_{con} features respectively. The objective is to train the main classifier F such that it accurately predicts y conditionally independent from the sensitive attribute z given the admissible features x_{adm}. Thus, the adversarial debiasing algorithm must only be applied on the contaminated features x_{con}. To control the debiasing procedure, the overall model structure is initialized as an ensemble of two components: (i) *Control Expert* (CE) that considers the admissible features x_{adm} and (ii) *Mitigation Expert* (ME) that takes the contaminated attributes x_{con}. Each expert model takes different input features so that each focuses on other patterns in the data. Other work has used this method to control known database biases (Clark, Yatskar, and Zettlemoyer, 2019). However, in this setup, only ME is affected by the adversarial learning game and enables the ensembled model to be trained simultaneously.

Unlike vanilla models that generate conditional distributions $p(y|x)$, the main classifier F models $p(y|h_{ce} \cap h_{me})$. The F maintains fairness such that predictions $\hat{y}$ and protected attribute z are conditionally independent given the admissible features x_{adm}. To achieve this objective, the CE transforms x_{adm} into representation h_{ce} where

ME encodes x_{con} and z into representation h_{me}. It should be noted here that z is used as an additional feature into the unstructured contaminated features x_{con} that derives representation $h_{me} = (x_{con}, z)$. By considering z as an additional input feature as part of x_{con}, the implicit and explicit bias would be amplified. This helps the adversarial algorithm to be more effective in mitigating the protected information from the encoded representation h_{me} without reducing the quality of the generated predictions significantly. The protected information in the admissible features x_{adm}, on the other hand, are present in a latent manner via the distribution of the observations. Thus, given representations h_{ce} and h_{me}, the main classifier or so-called *predictor* F models the distribution $q_f(y|h_{ce} \cap h_{me})$ to generate the main task predictions $\hat{y}$.

In order to make h_{me} oblivious from z while controlling the admissible associations through h_{ce}, an adversarial game is designed by introducing an *adversary* A. The adversary A is trained to predict z based on the encoded representation h_{me} and models $q_a(z|h_{me})$. Simultaneously, ME encodes representation h_{me} such that it minimizes the likelihood of inferring z correctly by the adversary. Thus, in an intuitive sense, representation h_{me} must be maximally informative for F to predict the main task labels while minimizing A's ability to predict z to ensure that $\hat{y}$ and z are conditionally independent given representation h_{ce}.

4.1.2 Optimization of the Adversarial Algorithm

The predictor F and adversary A are trained to predict y and z respectively by minimizing their cost functions. But, instead of representation h_{ce}, h_{me} is trained to make the adversary fail in predicting z. More formally, representations h_{ce} and h_{me}, predictor F and adversary A jointly optimizes the following adversarial minimax game:

$$\min_{h_{ce}, h_{me}, F} \max_{A} J(h_{ce}, h_{me}, F, A) \tag{4.1}$$

$$J(h_{ce}, h_{me}, F, A) = \mathop{\mathbb{E}}_{h_{ce}, h_{me}, z, y \sim p(h_{ce}, h_{me}, z, y)} [\gamma \log q_a(z|h_{me}) - \log q_f(y|h_{me} \cap h_{ce})] \tag{4.2}$$

Where γ controls the intensity of the debiasing effect in h_{me} from the adversary, $\log q_a(z|h_{me})$ and $\log q_f(y|h_{me} \cap h_{ce})$ are the objective functions and conditional entropies for F and A, and $p(h_{ce}, h_{me}, z, y)$ is the joint distribution with $h_{ce}(x_{adm})$ and $h_{me}(x_{con}, z)$. To prove the optimality of the minimax game, the optimal adversary A^* and optimal predictor F^* are deduced given fixed CE and ME.

Claim 1: Adversarial Minimax Game Optimum

Given a fixed CE and ME, the optimal adversary generates $q_a^*(z|h_{me}) = p(z|h_{me})$ and the optimal predictor outputs $q_f^*(y|h_{me} \cap h_{ce}) = p(y|h_{me} \cap h_{ce})$. Then, when γ is infinitely large, representation h_{me} is oblivious for z.

Proof. First, the optimal solution to obtain the stationary point for adversary q_a is proved which assumes convexity in the objective with respect to each distribution. Given a fixed CE and ME, the following optimization problem is defined:

$$\min_{A} \quad -J(h_{ce}, h_{me}, F, A) \qquad \text{s.t.} \quad \sum_{z} q_a(z|h_{me}) = 1, \forall h_{me} \tag{4.3}$$

Then, $\mathcal{L} = J(CE, ME, F, A) - \sum_{h_{me}} \lambda(h_{me})[\sum_z q_a(z|h_{me}) - 1]$ is the lagrangian dual function with $\lambda(h_{me})$ as the dual variables introduced for equality constraints. The optimal adversary satisfies the following:

$$0 = \frac{\partial \mathcal{L}}{\partial q_a^*(z|h_{me})} \tag{4.4}$$

$$\Leftrightarrow 0 = -\sum_{h_{ce}, h_{me}, z, y} \frac{\partial J}{\partial q_a^*(z|h_{me})} p(h_{ce}, h_{me}, z, y) - \lambda(h_{me}) \tag{4.5}$$

$$\Leftrightarrow 0 = -\sum_{h_{ce}, h_{me}, z, y} \frac{p(h_{ce}, h_{me}, z, y)}{q_a^*(z|h_{me})} - \lambda(h_{me}) \tag{4.6}$$

$$\Leftrightarrow 0 = -\sum_{h_{me}, z} \frac{p(z, h_{me})}{q_a^*(z|h_{me})} - \lambda(h_{me}) \tag{4.7}$$

$$\Leftrightarrow \lambda(h_{me}) q_a^*(z|h_{me}) = -p(z, h_{me}) \tag{4.8}$$

Given $\sum_z q_a^*(z|h_{me}) = 1$ and summing both sides with z, the following is obtained:

$$\lambda(h_{me}) = -p(h_{me}) \tag{4.9}$$

Then, substituting equation (4.9) back into equation (4.8), the A^* is derived as:

$$q_a^*(z|h_{me}) = \frac{p(z, h_{me})}{p(h_{me})} = p(z|h_{me}) \tag{4.10}$$

In the same setting, taking the partial derivative with respect to $q_f(y|h_{me} \cap h_{ce})$ and setting it equal to zero, it can be proved that $q_f^*(y|h_{me} \cap h_{ce}) = p(y|h_{me} \cap h_{ce})$. Thus,

by substituting q_a^* and q_f^* back into the minimax objective function (4.2), it can be rewritten as a minimization problem only with respect to ME and CE as follows:

$$\min_{h_{me},h_{ce}} J(h_{me},h_{ce}) = \mathop{\mathbb{E}}_{h_{ce},h_{me},z,y\sim p(h_{ce},h_{me},z,y)} [\gamma \log p_a(z|h_{me}) - \log p_f(y|h_{me} \cap h_{ce})] \quad (4.11)$$

When the optimum for A^* is $\log p_a(z|h_{me})$, the predictions in A^* are equal to $p_a(z|h_{me})$. To ensure that h_{me} is oblivious for z, $p_a(z|h_{me}) = p(z)$ must hold which is achieved in equation (4.11) with an infinite γ. This can be proved by considering the conditional entropy $H(z|h_{me})$ for A^* and the entropy $H(z)$ for z. By definition, $H(z|h_{me})$ in the adversarial game is formalized as:

$$H(z|h_{me}) = \sum_z p_a(z|h_{me}) \log p_a(z|h_{me}) \quad (4.12)$$

Given $p_a(z|h_{me}) > 0$ and $z \in (0,1)$, then (4.12) is maximized in (4.11) as:

$$0 = \frac{\partial H(z|h_{me})}{\partial p_a(z|h_{me})} \quad (4.13)$$

$$\Leftrightarrow 0 = \log p_a(z=0|h_{me}) + 1 - \log[1 - p_a(z=0|h_{me})] - 1 \quad (4.14)$$

$$\Leftrightarrow 0 = \log p_a(z=0|h_{me}) - \log[1 - p_a(z=0|h_{me})] \quad (4.15)$$

$$\Leftrightarrow 0 = \log \frac{p_a(z=0|h_{me})}{[1 - p_a(z=0|h_{me})]} \quad (4.16)$$

Then, the maximum for $H(z|h_{me})$ is obtained as:

$$\frac{p_a(z=0|h_{me})}{[1 - p_a(z=0|h_{me})]} = 1 \quad (4.17)$$

With $p_a(z|h_{me}) = 1/2$ for each z. To verify, substituting (4.17) back into (4.16) derives zero. Thus, when $p_a(z|h_{me}) = 1/2$ for $z \in (0,1)$, $H(z|h_{me}) = H(z))$ such that h_{me} is not informative for z. But, if h_{me} contains predictive information for z as:

$$\begin{aligned} p_a(z|h_{me}) = p(z) + \epsilon \quad &, z = 0 \\ p(z) - \epsilon \quad &, z = 1 \end{aligned} \quad (4.18)$$

With ϵ as a small positive, then $H(z|h_{me}) \neq H(z)$ and (4.12) is not maximized. However, since the overall objective function in (4.11) is minimized, the $H(z|h_{me})$ is maximized with infinite γ which satisfies $H(z|h_{me}) = H(z)$ or $p_a(z|h_{me}) = p(z)$. Thus, when $H(z|h_{me})$ is maximized in the adversarial minimax game, $p_a(z|h_{me}) = p(z)$ holds for which h_{me} and z become independent from each other. □

In theory, the entropy function is a measure of disorder where its value ranges between $[0, \infty]$. When the measure approaches zero, the predictive outcomes are considered to be accurate with a high certainty. On the other extreme, when the metric approaches ∞, the predictions are inaccurate or highly uncertain to occur (Goodfellow, Bengio, and Courville, 2016). Then, the average entropy value over all generated predictions approaches one or ∞ which can be considered to be equal to randomness. It should be noted that the average entropy value can be greater than one which depends on the number of prediction classes. Thus, when equation (4.11) is minimized with infinite γ, equation (4.12) is maximized such that $h_{me}) = H(z)$ or $p_a(z|h_{me}) = p(z)$ holds. Hence, representation h_{me} is oblivious for z when the optimum for adversary A^* is $\log p_a(z|h_{me})$ and γ is infinitely large.

4.1.3 Satisfying the Fairness Constraint

In Section 2.3, the fairness constraint that must be satisfied by the proposed adversarial ensemble-based framework is introduced. It was argued that when the main predictions $\hat{y}$ and protected attribute $z \in (0, 1)$ are conditionally independent given the admissible features x_{adm}, the following fairness constraint holds:

$$P(\hat{y}|z = 0, x_{adm}) = P(\hat{y}|z = 1, x_{adm}) \tag{4.19}$$

However, based on the proof for claim 1, the fairness constraint is not satisfied yet by the adversarial minimax game. Since predictor F consists of an ensemble of expert components and representation $h_{ce}(x_{adm})$ is not affected by the adversarial algorithm, it must be ensured that x_{adm} and z are uncorrelated to ensure conditional independence between predictions $\hat{y}$ and protected feature z given an admissible feature x_{adm}. Otherwise, $\hat{y}$ can still differ among the protected groups z in each x_{adm}. To achieve this objective, x_{adm} and z can be decorrelated by using sample weights that balances the distribution of z given x_{adm}.

Claim 2: Decorrelate the relationship between x_{adm} and z

When sample weights are derived as $w_i = \frac{1}{p(z|x_{adm})}$, the predictive outcome $\hat{y}$ and protected attribute z are conditionally independent given the admissible features x_{adm} such that $p(\hat{y}|z, x_{adm}) = p(\hat{y}|x_{adm})$ holds.

Proof. The conditional independence relationship among the predictive outcome $\hat{y}$, protected attribute z and admissible features x_{adm} can be formalized as:

$$\begin{aligned} p(\hat{y}|z, x_{adm}) &= \frac{p(\hat{y} \cap z|x_{adm})}{p(z|x_{adm})} \\ &= \frac{p(\hat{y}|x_{adm})p(z|x_{adm})}{p(z|x_{adm})} \\ &= p(\hat{y}|x_{adm}) \end{aligned} \tag{4.20}$$

To ensure that x_{adm} and z are uncorrelated, $p(z|x_{adm})$ must be uniformly distributed such that randomness is achieved. Given that all $p(z|x_{adm}) > 0$, the uniform distribution is obtained by the expected value for all z classes as:

$$\mathbb{E}[I(z = i)|x_{adm}] = \frac{1}{\mathbf{Z}} \quad \text{s.t.} \quad \sum_{i=1}^{z} \mathbb{E}[I(z = i)|x_{adm}] = 1 \tag{4.21}$$

With total $\mathbf{Z}$ values. Suppose $z \in (0, 1)$ and $p(z = 0|x_{adm}) < p(z = 1|x_{adm})$. Then, $p(z|x_{adm})$ is not uniformly distributed and equation (4.21) does not hold for all z.

To ensure that all $p(I(z = i)|x_{adm}) = \mathbb{E}[I(z = i)|x_{adm}]$, each z must be reweighted such that each value of z is equally likely given x_{adm}. By using inverse probability weighting, the weights for each z element are obtained as:

$$w_i(x_{adm}) = \frac{1}{\mathbf{Z} * p(I(z = i)|x_{adm})} \tag{4.22}$$

Given equation (4.22) and considering that each $I(z = i)$ conditioned on x_{adm} should not be predictive for z, the weighted expected value is derived as:

$$\mathbb{E}[w_i(x_{adm}) * I(z = i)|x_{adm}] = \frac{1}{\mathbf{Z}} \tag{4.23}$$

Given that all z values are mutually exclusive and summing all weighted $\mathbb{E}_w[I(z = i)|x_{adm}]$, the constraint in equation (4.21) is satisfied as:

$$\sum_{i=1}^{z} \mathbb{E}_w[I(z = i)|x_{adm}] = \sum_{i=1}^{z} \frac{1}{\mathbf{Z}} = 1 \tag{4.24}$$

When equation (4.24) holds, $p_w(z|x_{adm})$ is uniform with x_{adm} and z being independent from each other. Then, substituting the weighted probability distributions and expected value back into equation (4.20), the conditional independence relationship among $\hat{y}$, z and x_{adm} is proved as:

$$\begin{aligned} p(\hat{y}|z, x_{adm}) &= \frac{p_w(\hat{y} \cap z|x_{adm})}{\mathbb{E}_w[z|x_{adm}]} \\ &= \frac{p_w(\hat{y}|x_{adm})\mathbb{E}_w[z|x_{adm}]}{\mathbb{E}_w[z|x_{adm}]} \\ &= p_w(\hat{y}|x_{adm}) \end{aligned} \tag{4.25}$$

The inverse probability weighting is used to ensure that each z value is equally likely given x_{adm} such that $p(z|x_{adm})$ is uniform. In other words, x_{adm} and z become independent and uncorrelated from each other. Hence, the predictive outcome $\hat{y}$ and protected attribute z are conditionally independent given the admissible features x_{adm} such that $p(\hat{y}|z, x_{adm}) = p(\hat{y}|x_{adm})$ holds. □

Based on the proof for q_a^* and q_f^* in claim 1, it can be concluded that the adversarial minimax game ensures that information about z is mitigated from representation h_{me}. The optimal $q_a^*(z|h_{me})$ is maximized to ensure that representation h_{me} is oblivious from z, i.e. $p(z|h_{me}) = p(z)$, while the optimal $q_f^*(y|h_{me} \cap h_{ce})$ is minimized to obtain the best main task predictions. Moreover, to ensure that $\hat{y}$ and z are conditionally independent given x_{adm}, the proof for claim 2 has shown that a sample weighting scheme can be used. This ensure randomness and balances the distribution of $p(z|x_{adm})$ such that x_{adm} and z are uncorrelated. Hence, given the proofs for both claim 1 and claim 2, the fairness constraint in equation (4.19) holds in which acceptable discrimination can occur in the predictive outcomes $\hat{y}$ only through the dependency on the admissible features x_{adm}.

4.2 Extending the Backpropagation Weight Update-rules

In order to avoid that the gradients and updated weights in Mitigation Expert ME helps the adversary A in predicting the protected attribute z in the adversarial minimax game, the extended backpropagation weight update-rules are used as suggested by (Zhang, Lemoine, and Mitchell, 2018). As shown in Section 4.1, it has been proved that the adversarial minimax game in combination with sample weights ensures that the formulated fairness constraint is satisfied. In this setting, the predictor F and adversary A are both minimizing their entropy functions. Instead of the Control Expert (CE), the Mitigation Expert (ME) is optimized such that its representation $h_{me} = (x_{con}, z)$ is maximally informative for F and minimizes A's ability to predict protected attribute z. The Control Expert CE, however, uses sample weights to ensure that the acceptable discrimination in predictions $\hat{y}$ can occur only through

the dependency among the admissible features x_{adm}. In other words, CE, ME, F and A jointly optimizes equation (4.2) by updating the set of weights $[\theta_{ce}, \theta_{me}, \theta_f, \theta_a]$ in the model. As addressed in Section 2.1, the set of weights in a model are iteratively updated by taking the gradients of the loss with respect to the weights until the model converges at some optimum point. But, without constraining the gradients and updated weights in ME, the effectiveness of the adversarial learning algorithm could be affected. Therefore, the backpropagation weight update-rule for the set of weights is extended and can be formalized as:

$$\theta_{me} \leftarrow \theta_{me-1} - \eta\left(\frac{\partial L_f}{\partial \theta_{me-1}} - proj_{\frac{\partial L_a}{\partial \theta_{me-1}}} \frac{\partial L_f}{\partial \theta_{me-1}} - \gamma \frac{\partial L_a}{\partial \theta_{me-1}}\right) \tag{4.26}$$

$$\theta_{ce} \leftarrow \theta_{ce-1} - \eta \frac{\partial L_f}{\partial \theta_{ce-1}} \tag{4.27}$$

$$\theta_f \leftarrow \theta_{f-1} - \eta \frac{\partial L_f}{\partial \theta_{f-1}} \tag{4.28}$$

$$\theta_a \leftarrow \theta_{a-1} - \eta \frac{\partial L_a}{\partial \theta_{a-1}} \tag{4.29}$$

Where η is the learning rate and weight updates are derived by the partial derivatives of the F's loss L_f or A's loss L_a with respect to the set of weights at each training step. It can be seen that the weight update-rule for θ_{ce}, θ_f and θ_a follow the standard update procedure as introduced in Section 2.1. The weights θ_{me}, however, differs where the update-rule is extended as suggested by (Zhang, Lemoine, and Mitchell, 2018). To prevent the gradients in ME and A move into opposite directions and avoid lack of convergence to the optimum point, the middle term $proj_{\frac{\partial L_a}{\partial \theta_{me-1}}} \frac{\partial L_f}{\partial \theta_{me-1}}$ is added. This ensures that gradients of L_f with respect to θ_{me} are orthogonal to the gradients of L_a with respect to θ_{me}. In other words, this prevents ME's updated weights and gradients helping A decrease its loss. Appendix A shows the derivation of projection term in more detail.

5 Debiasing Performance on Product Review Data

This Section focuses on validating the extended adversarial learning method on a multi-class classification task. Section 4 have proved that the adversarial ensemble-based framework ensures that main predictions and protected attribute are conditionally independent such that acceptable discrimination can occur only through the dependency among admissible features. To validate the proposed debiasing method, the model is used to generate product rating class predictions without discriminating among gender given the product categories. Section 5.1 introduces the database and addresses the preprocessing steps. Afterwards, in Section 5.2, the model architecture and evaluation metrics used in the experiment are addressed. Finally, Section 5.3 presents the training procedure and experimental results.

5.1 Amazon Product Review Database

This thesis focuses on proposing an adversarial ensemble-based framework to control and mitigate unintended gender biases in (un)structured data representations. Section 4 have presented and proved the adversarial minimax game in combination with the sample weighting scheme and extended backpropagation weight update-rule. To validate the effectiveness of the debiasing approach, a reasonably large database is needed in which both the main task y and protected feature z labels are available. Therefore, for this experiment, the Amazon Review Database with 233.1 million reviews is used to predict product rating classes based on product reviews and other metadata features (Ni, Li, and McAuley, 2019). However, to reduce the computational time and required resources, a subset of the data with 75.26 million reviews is used in which all customers and product items have at least five reviews.

Besides having product ratings and unstructured customer reviews, the database must contain the protected feature. This sensitive attribute is considered to be *gender* with levels $z \in [male, female]$. As in most real-life applications, however, this information was not available in the data due to privacy concerns and regulations. Even though not collected, the protected feature is derived from the Amazon usernames that customers used by using a gender detector library [1]. This library collected birth

[1] https://pypi.org/project/gender-detector/

record data from the United States (US) and the United Kingdom (UK) across a number of years via databases that were released by the two countries. More specifically, in the US, the Social Security Administration has released records with name and gender by year of births between 1880-2011 whereas, in the UK, the Office of National Statistics made the same data publicly available for the years between 1996-2011[2]. However, only the name-gender combinations were considered that occurred with a minimum of 5 births in the US and 3 births in the UK. In total, the gender detector contains 101,749 unique names that were categorized into $[male, female]$. Before the protected attribute was derived from the Amazon usernames, the usernames were cleaned by removing numbers and punctuations. Also, since the library processed the names in lowercase letters, all username characters were set to lowercase letters. After using the gender detector, all observations that were not matched and labelled as *unknown* for gender were removed. Thus, the database used for the experiment was reduced to 5,416,688 observations.

Throughout this work, it was argued that it might be desirable to discriminate in the main predictions based on justified dependency among admissible information; while ensuring conditional independence with sensitive information. Therefore, the experimental database uses *Product Category* as the admissible feature since it is likely that this feature contains stereotypical gender associations. The feature's categories are formatted as dummy variables. Originally, the database contained 20 product categories which are shown in Appendix B.1. However, 10 product categories that are heavily skewed in the gender distribution are selected to be used in the experiment. This reduced the database to 3,575,999 observations. Table 5.1 shows the selected product categories. When the data is skewed towards some gender group, the model can naturally reflect these patterns and discriminate in the predictions as addressed in Section 2.4. However, as argued in Section 4.1.3, the discrimination in product rating predictions can be considered as acceptable when conditional independence with gender groups is ensured given the product categories. For instance, when male and female customers review products from the same category, it is justifiable to assume similarity in product interests. Therefore, the product rating predictions should not differ among gender which is considered to be unfair otherwise. But, when male and female customers review products from different categories, the difference in rating predictions is justifiable. Both the male and female customers are not comparable in terms of product interests for which the possible discrimination in ratings is acceptable. Moreover, the product category feature can contain predictive information for the main prediction task. When *beauty* category have on average a higher rating than *sports* category, a very positive review for beauty products would strengthen a rating prediction for 4- or 5-stars compared to reviews with a similar sentiment for the averaged lower rated sports products. The prediction results would, therefore, be more realistic.

[2] https://github.com/OpenGenderTracking/globalnamedata

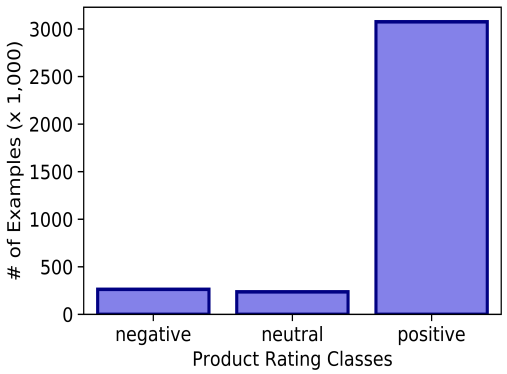

FIGURE 5.1: Data Distribution over Product Rating Classes

Category	Female	Male
All Beauty	73.5	26.5
Amazon Fashion	83.1	16.9
Arts and Crafts	80.6	19.4
Automotive	9.0	91.0
Scientific	11.6	88.4
Luxury Beauty	75.2	24.8
Musical Instruments	10.8	89.2
Sports	20.9	79.1
Tools and Home	17.1	82.9
Video Games	20.6	79.4

TABLE 5.1: Categories with Gender in %

Since the aim of the experiment is to classify customer reviews with the specific product category in rating classes, the dependent variable is derived from the *star-rating* feature in the database. The star-rating attribute, however, is factored and releveled from a one- to five-star range to three levels: (i) *negative* rating which is obtained by grouping all one- and two-star ratings, (ii) *neutral* rating which is based on all three-star ratings and (iii) *positive* rating which is formed by combining all four- and five-star ratings. Figure 5.1 shows the class distributions over the product rating categories which indicates that the database is highly imbalanced. Since the positive ratings can be considered to be the majority class, the minority negative and neutral classes are harder to predict due to the imbalanced distribution. This means that the model is challenged to learn from and differentiate the observations from the minority classes from the majority class. To ensure that the prediction results are reliable and after a training set containing two-thirds of the entire database is created, the *class reweighting* method is used instead of applying sampling methods. When sampling methods are used, it might be possible that the gender distribution in the product categories are affected. However, this is not desired since the objective is to capture the stereotypical associations to mimic the real-world context as closely as possible. In this setting, the observations from the positive rating class gets a weight assigned that is relatively lower compared to negative and neutral rating classes. Then, during training, the class weights are multiplied by the loss value for each training example such that the overall model loss focuses more on the minority classes instead of the majority positive rating class. This ensures that all product rating classes are equally considered in the training procedure. In the adversarial ensemble-based framework, the relationship between product categories and gender information must be decorrelated. Section 4.1.3 has introduced the *sample reweighting* method which is derived from the training set. These sample weights are applied similarly as the class weights.

During the data preprocessing and model training phase, the validation and test sets are untouched and created by splitting the remaining one-third of the database evenly. Where the validation set is used for the model tuning procedure, the test set is used to evaluate the prediction performance on unseen data. As mentioned, the obtained database contains 3,575,999 examples. To further reduce the computational time and required resources, a random sample of 50,000 observations is obtained from the database such that the gender distribution in each product category and product rating class distribution is proportionally maintained. Thus, on this basis, the database that is used in the experiment contains 50,000 observations with each related to four variables: (i) product reviews in text, (ii) product category for the reviewed product which contains 10 categories as shown in Table 5.1, (iii) gender which indicates whether the customer is a male or female and (iv) product rating class which are categorized into negative, neutral and positive rating classes.

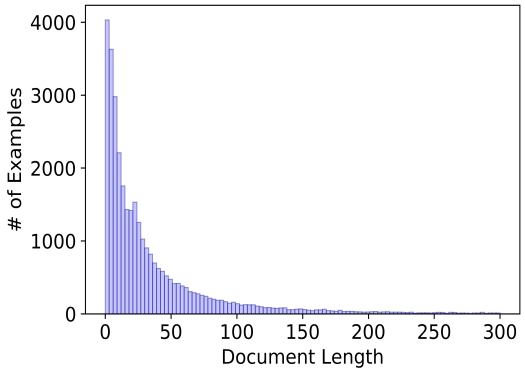

FIGURE 5.2: Doc. Length Distribution

Length (in <=)	# of Examples
50	25,890
100	30,196
150	31,693
200	32,383
250	32,735

TABLE 5.2: Number of Examples per Doc. Length

In Section 2.4, it was addressed that some studies have found that text representations can contain gender biases. These sensitive latent information could be learned by the learning algorithm through the encoded data representations (Chen et al., 2019). Therefore, to validate the extended adversarial learning algorithm, the customer reviews are considered to be the contaminated features in this thesis. The product reviews are provided in the unstructured form of natural human language. The raw review documents, however, cannot directly be used to build the model for which the text must be cleaned. To process the raw text, all upper-case characters were set to lower-case letters and numbers, punctuations, non-ASCII characters and extra whitespaces were removed. All contractions and abbreviations, however, were converted to their respective meaning. The stopwords are not removed in the text documents. These terms can affect the context and meaning of specific words and eventually influence the predicted rating classes. For instance, the short review document "I do not like this product so I do not recommend it" would most likely obtain a negative predicted rating compared to "like product recommend" in which all stopwords are removed and most likely obtains a positive predicted rating. Moreover, the text documents are formatted in sequences of words. More specifically, the processed words in each document are tokenized into sequences of

tokens. Then, a vocabulary is created based on the training set such that all words with a minimum frequency of 2 in all documents are collected. Here, it is assumed that all words with a frequency of 1 are imaginary or non-informative words. Thus, by only considering terms that occur twice at minimum, the computational time and required memory capacity can be reduced. Since pretrained 300-dimensional GloVe Embeddings are used, it is essential to create a vocabulary with enough words to ensure that most terms can be captured by the pretrained vectors. Section 5.2 explains the GloVe Embeddings in more detail. As a result, the vocabulary consists of 17,641 unique words. Then, word tokens in each document is substituted with the related index where non-occurring terms get an index assigned that represents an out-of-vocabulary token. Finally, each review must have the same length for which the sequences are padded and truncated. As shown in Figure 5.2 and Table 5.2, the gap in the number of observations between the document lengths reduces. Since the train set contains 33,500 examples and most examples are captured, the sequences are padded and truncated such that each document length is equal to 150 terms.

5.2 Model Architecture and Evaluation Measures

Before the extended adversarial learning algorithm can be tested, the model architecture and evaluation measures must be specified. As addressed in Section 4.1 and shown in Figure 5.3, the overall model is structured as an ensemble of two components: (i) *Control Expert* (CE) and (ii) *Mitigation Expert* (ME) that creates representations $h_{ce} = (x_{adm})$ and $h_{me} = (x_{con}, z)$ respectively. The predictor F generates the main task predictions $\hat{y}$ based on the encoded representations h_{ce} and h_{me}. While the admissible features x_{adm} is decorrelated with the protected attribute z via the sample weighting scheme, ME is initialized as a multi-output model. The transformed hidden representation h_{me} is fed into the predictor F to predict y and adversary A to predict z. This setup makes it possible to implement the adversarial debiasing algorithm and make h_{me} oblivious from z as discussed and proved in Section 4.1.2.

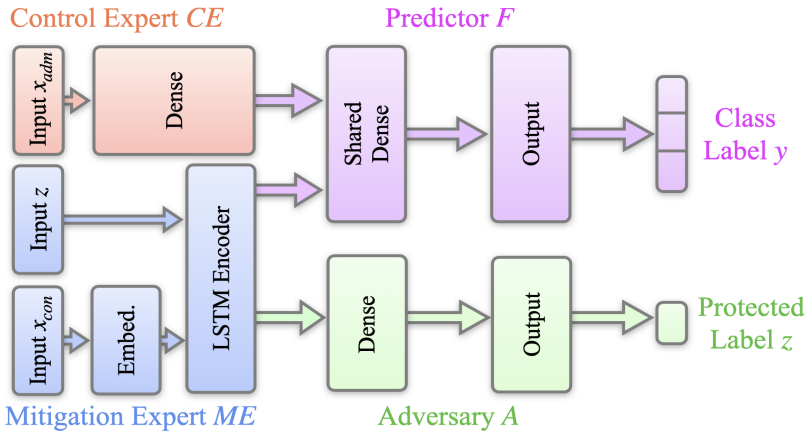

FIGURE 5.3: Model Architecture used in the Experiment

5.2.1 Control Expert

The Control Expert CE takes the structured *product category* features as input - which are considered to be the admissible features x_{adm} - is modeled as a feed-forward network. As addressed in Section 2.1, this type of design is an interconnected structure in which layers and nodes are only connected into one direction. Each node receives input from preceding nodes and generates output values for subsequent nodes until the output prediction is generated (Goodfellow, Bengio, and Courville, 2016). More specifically, the architecture consists of two layer components. The input layer that takes x_{adm} and has a number of nodes which is set equal to the 10 product category dummies. Before the transformed representation can be ensembled in predictor F, the hidden layer component can be tuned. In theory, when multiple features are fed into a network, interactions among input features can occur for which the most optimal architecture with the best prediction performance needs to be obtained. This can be achieved by tuning and selecting the optimal number of nodes and hidden layers. However, in this work, there are 10 mutually exclusive dummies that are fed into CE. Thus, it is expected that there are no interactions among dummies. Still, to validate that classification performance does not improve, the number of hidden nodes and layers are tuned. As (Goodfellow, Bengio, and Courville, 2016) argued, the optimal number of nodes is usually derived by trial-and-error between the number of nodes in the input n_{inp} and output n_{out} layer. Therefore, in this work, the number of nodes in the hidden component is tuned by iterating over three equally spaced values in the interval $[n_{out}, n_{inp}]$. More specifically, since the main task prediction in the experiment is focused on classifying examples into three classes, the hidden nodes are iteratively tuned over $[4, 7, 10]$ to verify whether a minimum, average or maximum number of hidden nodes are needed. Even though more than one hidden layer rarely improves the model performance, it must be validated that the performance does not improve with more hidden layers (Saeed and Snášel, 2014). Therefore, two variants for CE are initialized: variant (i) contains one hidden layer and variant (ii) has two hidden layers. The number of hidden nodes in both variants are tuned over the same set of $[4, 7, 10]$. Then, after transforming x_{adm}, the representation from the last hidden component h_{ce} is fed into predictor F.

Since F focuses on a classification problem with three classes, the output layer in F is initialized with the *softmax* activation function. Section 5.2.3 addresses F's architecture in more detail. However, one limitation of the softmax function is the potential occurrence of the *saturation problem*. This problem can occur when sigmoidal functions, like softmax, are used. The softmax function compresses activation values into a space between $[0, 1]$ where the derived probabilities over the output nodes sum to one. The gradients around the boundaries, however, approaches zero. Small gradients imply that the weights of the initial layers will not be updated effectively with each training step, and can lead to overall inaccuracy of the network (Goodfellow, Bengio, and Courville, 2016; Pascanu, Mikolov, and Bengio, 2013). To prevent

this problem, the activation values in the hidden layers in CE are derived with the *rectified linear unit*, i.e. *ReLU*, activation function which can be formalized as:

$$\phi(z_j) = \max(0, z_j) \tag{5.1}$$

With z_j as the weighted activation value from the preceding layer where it returns the weighted value if it is equal or above zero; otherwise the function returns zero. This activation function avoids the saturation problem because its function and gradients are strictly monotonic compared to the sigmoidal functions; where its gradients are non-monotonic (Goodfellow, Bengio, and Courville, 2016).

5.2.2 Mitigation Expert

Unlike the Control Expert CE that considers the admissible features x_{adm}, the Mitigation Expert ME takes the contaminated features x_{con} from which all protected information must be mitigated. The ME encodes the contaminated x_{con} and protected attributes z into representation h_{me} which is fed into predictor F and adversary A. In this setting, the adversarial minimax game is implemented to make h_{me} oblivious from z. However, in the experiment, the contaminated features x_{con} are considered to be the *customer reviews* which can be viewed as a sequence of words whereas the protected gender feature z can be defined in clear structured groups. Therefore, representation h_{me} is derived by using a *Long Short-Term Memory (LSTM)* module with pretrained GloVe word embeddings[3]. Instead of passing the text features straight through the network without considering the order, this module views the input data as a sequence (Goodfellow, Bengio, and Courville, 2016; Minaee et al., 2020). This is preferred over the feed-forward architecture since word dependencies and text structures can be captured in more detail via sequential ordering (Sutskever, Martens, and Hinton, 2011). As argued in Section 4.1.1, the protected attribute z is used as an additional feature into x_{con} to amplify the implicit bias. Also, it would help the adversarial learning algorithm to be more effective in omitting the sensitive information from h_{me} without reducing the quality of the generated predictions with a significant number. In this context, the sensitive attribute z is added as an extra dimension in the GloVe embeddings to accentuate the latent gender association among words. Hence, the 300-dimensional GloVe embeddings are expanded to be 301-dimensional in the model. The method and architecture are explained below:

Recurrent Neural Networks The LSTM model is a variant of a *Recurrent Neural Network (RNN)*. To understand the mechanisms of LSTM networks, it is a good start to first evaluate RNNs. Intuitively, RNNs receive an input, updates its hidden states and generates predictions. Figure 5.4 shows the mechanism in a so-called *RNN cell*. In contrast with feed-forward networks, RNNs contain multiple RNN cells or so-called *recurrent units* that incorporate a feedback loop through their connection. The

[3]https://nlp.stanford.edu/projects/glove/

number of recurrent units, however, depends on the variable length of the input sequence or so-called *timesteps* in the sequences (Elman, 1990). Thus, the RNN can be seen as a rich and dynamic model with memory that iterates over the sequence before generating a prediction. The feed-forward network is not able to learn this type of information (Goodfellow, Bengio, and Courville, 2016).

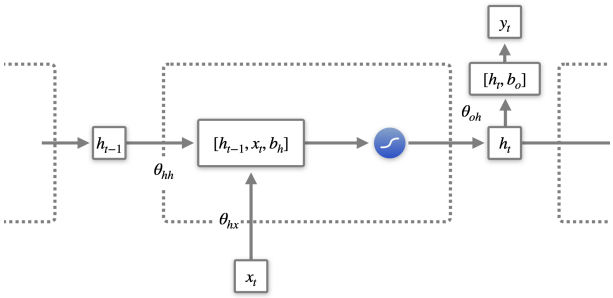

FIGURE 5.4: Variable flow in RNN cell

The RNN can be formalized as follows: consider the database that contains samples $x \in X$ with unstructured text documents $x_{u1}, \ldots, x_{un}$. Each x_{ui} is preprocessed into a sequence of vectors or so-called word tokens $x_1, \ldots, x_t$. Then, the RNN computes a sequence of hidden states $h_1, \ldots, h_t$ and a sequence of outputs $y_1, \ldots, y_t$ by iterating over the input sequence from $t = 1$ to T with the following equations:

$$h_t = \phi(\theta_{hx} x_t + \theta_{hh} h_{t-1} + b_h) \tag{5.2}$$

$$y_t = \theta_{oh} h_t + b_0 \tag{5.3}$$

Where the input-to-hidden weight matrix θ_{hx} is multiplied by vector x_t, summed with a multiplication of the hidden-to-hidden weight matrix θ_{hh} and the hidden state from the previous timestep h_{t-1}, and summed with a bias vector b_h. To obtain the current hidden state, the computed value is transformed by using the *tanh* function. Then, to derive the output at timestep t, the current hidden state h_t is multiplied by the hidden-to-output weight matrix θ_{oh}, summed with a bias b_o and transformed with some activation function. This is equivalent to the output layer in feed-forward networks. At $t = 1$, however, the $\theta_{hh} h_{t-1}$ is initialized with a bias vector h_{init} since h_{t-1} is undefined (Goodfellow, Bengio, and Courville, 2016). This RNN generates an output at each timestep which is referred to as so-called *many-to-many* architecture (Goodfellow, Bengio, and Courville, 2016). In this thesis, however, a *many-to-one* architecture is considered in which the sequences with word tokens are transformed into a single output prediction. This output is equivalent to the hidden state value from the last recurrent unit which is defined as representation h_{me}.

When training RNNs, an extension of the backpropagation algorithm - as addressed in Section 2.1 - is used which is referred to as *backpropagation-through-time* (Goodfellow, Bengio, and Courville, 2016). In this case, the ordered sequences of computations connected through each timestep defines *time* (Goodfellow, Bengio, and Courville, 2016). However, in practice, the training procedure can be highly unstable since it can experience the *vanishing gradient problem* (Bengio, Simard, and Frasconi, 1994). Because recurrent units are related through multiplication, the gradients can explode or vanish when frequently multiplied by gradients slightly greater or less than one (Bengio, Simard, and Frasconi, 1994). Even though the exploding gradients can be easily solved by truncating or squashing the gradients, the vanishing gradients can prevent the model from learning and are harder to solve in RNNs. Thus, RNNs can experience difficulties when learning long-term dependencies is required.

Long Short-Term Memory Networks One approach to deal with the vanishing gradient problem is to modify the model by including *memory units* that can handle long-term dependencies and memorization. The LSTM architecture contains such memory units as proposed by (Hochreiter and Schmidhuber, 1997). Therefore, the contaminated text documents x_{con} in ME are modeled via a LSTM module to ensure word dependencies can be learned. Figure 5.5 shows the mechanism in a *memory cell*. In essence, LSTMs use gated cells to control information such that the errors are maintained over many timesteps. This avoids the vanishing gradients problem. Moreover, gated cells can block or pass information based on their strength and are filtered with their trainable weights (Hochreiter and Schmidhuber, 1997).

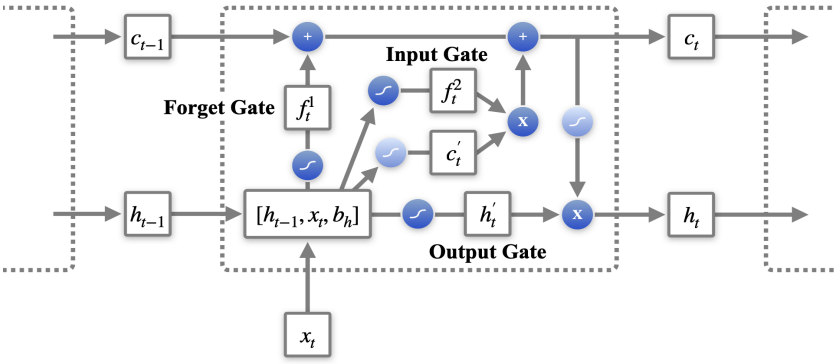

FIGURE 5.5: Variable flow in Memory cell

When the same setting is considered with the database that contains samples $x \in X$ with each text document x_{ui} tokenized into sequences of vectors $x_1, \ldots, x_t$, the LSTM iterates over the sequences from $t = 1$ to T as follows:

$$f_t^1 = \sigma(\theta_{f^1}[h_{t-1}, x_t] + b_{f^1}) \tag{5.4}$$

$$f_t^2 = \sigma(\theta_{f^2}[h_{t-1}, x_t] + b_{f^2}) \odot \tanh(\theta_{f^2}[h_{t-1}, x_t] + b_{f^2}) \tag{5.5}$$

$$h_t^{'} = \sigma(\theta_{h_t'}[h_{t-1}, x_t] + b_{h_t'}) \odot \tanh(c_t) \tag{5.6}$$

Where (5.4) is the *forget gate* that filters long-term information to remember by taking a *sigmoid* function over weight matrix θ_{f_1}, concatenated hidden state from the preceding cell h_{t-1} and input vector x_t, and some bias b_{f_1}. Equation (5.5) is the *input gate* that processes the new information into the cell state by considering again a *sigmoid* function over weight matrix θ_{f2}, concatenated hidden state from the preceding cell h_{t-1} and input vector x_t, and some bias b_{f_2}. But, instead of streaming this value into the cell's current state, the value is multiplied by a *tanh* function with the same parameters. Then, (5.4) and (5.5) are summed to derive the cell's current state c_t. This summation maintains a constant error through the long-term sequence during training and solves the vanishing gradient problem (Goodfellow, Bengio, and Courville, 2016). Equation (5.6) is the *output gate* that computes the hidden state by taking a *sigmoid* function over weight matrix $\theta_{h_t'}$, concatenated hidden state from preceding cell h_{t-1} and input vector x_t, and some bias $b_{h_t'}$. This is multiplied by *tanh* function with the cell's current state c_t. The output gate, however, is an equivalent of the recurrent unit in the simple RNN (Goodfellow, Bengio, and Courville, 2016).

As addressed, this thesis considers a many-to-one architecture for ME. The LSTM output is equivalent to the hidden state value from the last memory cell h_{me} which is fed into predictor F and adversary A. The module is initialized with all biases equal to zero at the start of the training procedure; except for the bias in the forget gate which is initialized as $b_{f_1} = 1$. (Jozefowicz, Zaremba, and Sutskever, 2015) have shown that the performance improves when a bias of 1 is used in the forget gate of every memory cell. The default activation functions - as mentioned in the formal explanation - are used as suggested by (Hochreiter and Schmidhuber, 1997; Jozefowicz, Zaremba, and Sutskever, 2015). In feed-forward networks, the number of hidden nodes and hidden layers must be determined. However, in LSTMs, the hidden nodes are referred to as the units in the memory cell which are initialized equal to the dimension of the input vectors x_t to capture all available information in input tokens (Elazar and Goldberg, 2018; Goodfellow, Bengio, and Courville, 2016; Sutskever, Martens, and Hinton, 2011). As addressed in Section 5.1, the experiment uses 300-dimensional GloVe embeddings which are expanded to 301-dimensional vectors by including the protected attribute z to each vector. Therefore, each memory cell is initialized with 301 units. The number of memory cells, however, can be referred to the number of time steps taken by the LSTM component through each sequence. As determined in Section 5.1, all sequences are padded and truncated to be of length 150 in the experiment. Thus, the number of memory cells are set to

contain 150 cells. Moreover, the *ME* is initialized with one LSTM layer and is not tuned with more layers for computational reasons.

Since LSTMs maintain information through memory cells, the *overfitting* problem can occur. This problem refers to a model that fits the training data almost perfectly. Specifically, overfitting occurs when models capture the noise in the training data for which the performance cannot generalize on new data (Goodfellow, Bengio, and Courville, 2016). To avoid this problem in the LSTM module, the *dropout* regularization method is implemented between the embedding and LSTM module. This probabilistically ignores nodes from the prior layer during training such that the weight contributions to subsequent layers are temporally removed on the forward pass and no updates are applied on the backward pass (Goodfellow, Bengio, and Courville, 2016). To implement dropout, the dropout rate is tuned over the set $[0, 0.2, 0.5]$ where the rate determines the percentage of nodes to ignores during training.

GloVe Word Embeddings Before the customer reviews x_{con} can be fed into the LSTM module, the words in each document must be transformed into word vectors via an embedding layer. Though, it is common practice to use pretrained embeddings to leverage the semantics and syntactic relationships learned from a large text corpus. However, as addressed in Section 2.4, some studies have identified that such pretrained embeddings contain stereotypical gender biases. Since the presents of gender biases in GloVe embeddings are more studied compared to other methods, the pretrained GloVe embeddings with 300-dimensional vectors is used in this work. But, what are word embeddings? Embeddings represent words w in a d-dimensional vector $w \in \mathbb{R}^d$ where words with similar semantic meaning obtain closely related vectors (Mikolov et al., 2013; Pennington, Socher, and Manning, 2014). Thus, the semantic information in vectors enables downstream models to learn the relationships among words (Allen and Hospedales, 2019; Mikolov, Yih, and Zweig, 2013).

The GloVe embeddings are trained over a Common Crawl corpus with 840 billion tokens and vocabulary size of 2.2 million to create 300-dimensional word vectors. Intuitively, this language modeling method has learned the word embeddings via a log-bilinear model which takes a term co-occurrence matrix (TCM) based on a skip-gram window as input (Pennington, Socher, and Manning, 2014). The skip-gram window determines the considered size of the local context for a specific target word, i.e. focal word. In other words, the GloVe model is trained to predict and transform the focal word into a word embedding based on given contextual terms (Pennington, Socher, and Manning, 2014). To use the pretrained embeddings, this work uses the word vectors as weights in the embedding layer. This is often referred to as *transfer learning*. To use the pretrained embeddings for other prediction domains, it is common to retrain or so-called *fine-tune* the vectors during the training procedure. However, for computing power reasons, the GloVe embeddings are not fine-tuned in this thesis. Even though not retrained, it would be expected that fine-tuning the

pretrained vectors amplifies and captures the stereotypical gender biases in more detail which should be kept in mind for future studies.

5.2.3 Classification Components

In the experiment, the two expert components CE and ME are simultaneously trained and combined via a shared hidden layer in predictor F to generate the main predictions $\hat{y}$. The shared hidden layer concatenates the experts' activation values as:

$$h_F = F_\delta(h_{ce}(x_{adm}), h_{me}(x_{con}, z)) \tag{5.7}$$

With h_F as the transformed hidden representation for the shared hidden layer in F and F_δ as the weight matrix and activation function to derive representation h_F. Predictor F is designed as a feed-forward module with one hidden and output layer. As addressed in Section 5.2.1, the number of hidden nodes and layer components can affect the performance. However, when a model contains more than one hidden layer, it rarely improves the overall performance as (Saeed and Snášel, 2014) have argued. Additionally, when more hyperparameters are tuned via the grid search, the number of training models increases. Therefore, given these reasons, predictor F is initialized with one hidden layer component. The number of hidden nodes, however, is tuned by the same principle as addressed in Section 5.2.1. More specifically, the number of nodes are tuned by iterating over three equally spaced values in the interval $[n_{out}, n_{inp}]$ or $[107, 209, 311]$. The number of output nodes, however, is equivalent to the number of classes in the classification problem at hand. In this work, the experiment is focused on predicting product rating classes which are divided into three classes. Hence, the number of output nodes is set to three nodes. Since the output layer has three nodes and focuses on a classification problem, the layer is set to use the *softmax* activation function. This is a sigmoidal function that distributes the predicted probabilities throughout each output node (Goodfellow, Bengio, and Courville, 2016). When a binary classification problem is addressed, the *softmax function* is equivalent to the *sigmoid function* (Goodfellow, Bengio, and Courville, 2016). The hidden layer considers the ReLU activation function given the fact that the saturation problem could occur as explained in Section 5.2.1.

Representation h_{me} from the Mitigation Expert (ME), however, is not only fed into predictor F. To implement the adversarial debiasing algorithm, h_{me} is also fed into adversary A to make h_{me} oblivious from protected attribute z. In terms of model architecture, adversary A is initialized with the same initialized design as used for predictor F. But, the number of nodes in the hidden layer component is not tuned over three equally spaced values in the interval $[n_{out}, n_{inp}]$. The A follows the same number of hidden nodes proportionally. More specifically, when the F trains a model with the minimum number of hidden nodes, i.e. 107 hidden nodes, the A uses also the minimum number of hidden nodes. This holds for the average and maximum

sets of hidden nodes as well. Hence, A uses the set $[101, 201, 301]$ as the minimum, average and maximum number of hidden nodes respectively. This ensures that both the F and A have the same prediction capacity during the adversarial game. Also, it reduces the number of training models in the grid search. Moreover, the number of output nodes is set to the number of classes in protected feature z. Since experiment focuses on debiasing representation h_{me} with respect to *gender*, the number of output nodes in the A is set to two which is equivalent to the binary $[male, female]$ classes. With respect to the activation functions, the *sigmoid function* is used in the binary output layer whereas the hidden layer is initialized with *ReLU* function.

During training, F and A are trained to minimize some loss function. In this work, the *categorical cross-entropy* and *binary cross-entropy* loss functions are used. These are selected because the experiment addresses a multi-class problem in the main task and binary problem in the adversarial task. The loss functions can be formalized as:

$$L(\hat{y}, y) = -\sum_{i=1}^{N}\sum_{K=1}^{K} y_{ik} \log F_k(h_{ce}(x_{admi}), h_{me}(x_{coni}, z_i)) \tag{5.8}$$

$$L(\hat{z}, z) = -\sum_{i=1}^{N}(z_i \log A(h_{me}(x_{coni}, z_i)) - (1 - z_i) \log(1 - A(h_{me}(x_{coni}, z_i)))) \tag{5.9}$$

Where y_{ik} is the ground-truth label for class k and i^{th}-example multiplied by the logarithmic probability prediction of predictor F for each class k and i^{th}-example in (5.8). In (5.9), z_i is the ground-truth label for each i^{th}-example in the sensitive prediction task. More intuitively, both cross-entropy measures how far away the prediction is for each class from the true target classes, which is either zero or one, and averages the error over the total number of training observations to obtain the overall loss (Goodfellow, Bengio, and Courville, 2016). Moreover, all weights in the model are being updated by using the *Adam* optimization method. As discussed in Section 2.1, this optimization approach avoids local minima and saddle points, and converges fast where only the initialized learning rate and mini-batch sizes can be tuned. To obtain the best performing model, the learning rate (LR) is initialized with the default $LR = 0.001$ in combination with a LR scheduler that reduces the LR by a factor of 0.1 when the training loss value has stopped improving after five epochs (Kingma and Ba, 2015). The mini-batch sizes are tuned by considering the sizes $[64, 128, 256]$. (Keskar et al., 2017) have shown that batch sizes outside of $[32, 512]$ are most likely to be sub-optimal for which the three equally spaced batch sizes $[64, 128, 256]$ are used in training. Moreover, to avoid overfitting and reduce the computational time, *early stopping* is used in this work. As discussed in Section 5.2.2, dropout is a regularization method that can be used to avoid overfitting. Early stopping, on the other hand, is another regularization method. This interrupts the training procedure when the validation loss value does not improve for a number of epochs (Goodfellow, Bengio,

and Courville, 2016). Hence, in the experiment, early stopping is used such that it interrupts the training procedure after 15 epochs without improvements on the validation loss. It is important to note that this method is only applied on the baseline model that contains ME, CE and F as introduced in Section 5.2.4. The adversarial debiasing setting, on the other hand, does not use early stopping during the training procedure. This would allow the adversarial minimax game to have more time in learning and mitigating the protected attribute z from representation h_{me}. Thus, in this work, the expert components ensembles in F where h_{me} is also fed into A. In the adversarial setting, both classifiers are simultaneously trained to predict $\hat{y}$ and $\hat{z}$ by minimizing the loss functions while the adversarial minimax game attempts to satisfy the desired level of fairness in predictor F.

5.2.4 Evaluation Setting and Measures

To evaluate the performance and the effectiveness of the extended adversarial learning algorithm, it is important to compare some measures among a baseline and debiased system. Therefore, two models are created and trained: (i) *biased model* and (ii) *debiased model*. The *biased model* serves as a baseline and follows the same architecture as previously introduced. However, instead of training the entire system simultaneously, the baseline does not consider the adversary A. In other words, the biased system trains the Control Expert CE, Mitigation Expert ME and predictor F to obtain the best performance on the main prediction task without being debiasing through A. This baseline setup ensures that the optimal hyperparameters are selected based on the main task performance in F. The *debiased model*, on the other hand, considers the same model components including A and implements the adversarial debiasing method as suggested in Section 4. However, instead of training the debiasing model from start, this system uses the pretrained weights from the baseline for CE, ME and F. This makes it possible to evaluate the effectiveness of the adversarial minimax game and sample weighting scheme in more detail, and makes the comparison more reliable. Moreover, as addressed in Section 4.1.2, the adversarial minimax game contains γ that controls the intensity of the debiasing effect. This trade-off parameter is, however, not tuned. Instead, the debiasing model increments γ over time as $\gamma = \sqrt{t}$ where t is the train step counter. (Zhang, Lemoine, and Mitchell, 2018) have shown that F experiences much easier time learning to make A fail in predicting the protected attribute when the debiasing parameter is increased as $\gamma = \sqrt{t}$. Also, the unintended gender bias that is addressed in the experiment must be removed completely. It is not justifiable to maintain some bias in the model since the protected groups can still be treated differently in that case. Hence, when γ increments over time, the debiasing results are more ethically sound.

To compare the performances among the models, two measures are used. Since the database is imbalanced in the target class distribution, the following class-specific measures are used that can identify the class- and overall performance:

Recall This measures the correctly classified examples compared to the total number of examples that actually belong to the specific class, where its value is bounded between $[0, 1]$ (Tharwat, 2018). The metric is for m classes and with true positive (TP) and false negative (FN) derived as:

$$Recall = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} TP_i + \sum_{i=1}^{m} FN_i} \tag{5.10}$$

Balanced Accuracy (BA) This computes the average accuracy rate per class between $[0, 1]$ which is more robust for imbalanced classes than the conventional accuracy metric. The conventional measure may yield an optimistic result for over-represented classes where the balanced accuracy will drop to chance when this occurs (Brodersen et al., 2010). The formula to derive this metric for m classes and with true positive (TP), false positive (FP), true negative (TN) and false negative (FN) is:

$$BA = \frac{1}{2}[\frac{TP_m}{TP_m + FN_m} + \frac{TN_m}{TN_m + FP_m}] \tag{5.11}$$

Fairness measures As introduced in Section 2.3, the fairness constraint that must hold after applying the adversarial ensemble-based framework is defined as:

$$P(\hat{y}|z = 0, x_{adm}) = P(\hat{y}|z = 1, x_{adm}) \tag{5.12}$$

This constraint requires that the proportion of each gender group obtain equal predictive outcomes per class given some admissible feature. To assess the effectiveness of the debiasing method, the Positive Rate (PR) is used which is derived as:

$$PR_m = \frac{TP_m + FP_m}{TP_m + FP_m + TN_m + FN_m} \tag{5.13}$$

With True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN) for each m class. This metric is computed based on predictions given some pre-selected product categories and assessed for each gender group separately. Thus, the Positive Rate (PR) evaluates whether the fairness constraint in (5.12) holds which is the main objective in the experiment.

5.3 Experimental Results

Before the effectiveness of the adversarial method can be assessed, the baseline and debiased models must be trained. The first step is to tune the baseline system to obtain the best architecture on the main prediction task. As introduced in Section 5.2.4, the model does not consider the adversary A for which only the Control Expert CE, Mitigation Expert ME and predictor F are considered. Section 5.2.1 have

addressed that two variants of CE must be initialized: variant (i) containing one hidden layer and variant (ii) with two hidden layers. Both variants are setup with the same parameters in ME and activation functions as explained in Section 5.2 to minimize the categorical cross-entropy cost function. The two variants are tuned over the remaining hyperparameters such as (i) number of hidden nodes for each hidden layer in CE and F iterates over $[4, 7, 10]$ and $[107, 209, 311]$, (ii) dropout rates in ME iterates over $[0, 0.2, 0.5]$ and (iii) mini-batch sizes in sets of $[64, 128, 256]$. In total, 162 models are trained with 100 epochs each where early stopping is used to interrupt the training procedure given the non-improving validation loss criteria. Moreover, the Adam optimizer is used with learning rate (LR) $= 0.001$ and LR scheduler that factors the LR by 0.1 when the training loss has stopped improving for 5 epochs.

After tuning the hyperparameters, all model performances are evaluated by assessing the loss curves on the training and validation data. This has shown that all systems with two hidden layers in CE either overfits the training data or have diverged immediately from start. The models with one hidden layer, on the other hand, resulted in six models that have converged. To validate these performances over the set of hyperparameter values, some other training methods should be tested in the learning setup. A simple method is to use *multiple random weight initializations* at the start of the learning procedure; where each random start ensures that the model is trained via different learning patterns. Another approach is to use different *optimization methods*; where each method adjusts the weights and learning rates differently. Also, the *learning rate* can be tuned with smaller values to reduce the learning speed and step sizes at each weight update iteration. This could avoid overfitting because it ensures that the updated weights are less likely to overshoot the optimum. It should be noted that smaller learning rates are more likely to derive at local minima and saddle points for which the learning rate must be initialized with some caution. Finally, different regularization methods can be tested. In the experiment, *dropout* is applied on the LSTM component. However, another regularization technique that can be used is *weight decay*. This shrinks the weights at every training iteration by multiplying the parameters with a factor slightly less than one (Goodfellow, Bengio, and Courville, 2016). Hence, these methods could be used to validate whether the six models that have converged are indeed the best and only model states with generalizable performances. But, due to computational reasons, this work has only applied the training setup as explained in Section 5.2.

From the six models that have converged, the two models with the smoothest learning curves - which refers to a stable training procedure - are used for further evaluation. Appendix C.1 shows the learning curves for both systems. One of the two models - which is defined as variant A - contains 4 and 311 hidden nodes in CE and F respectively, dropout rate of 0.2, and mini-batch size of 64. The other model - which is defined as variant B - contains 10 and 209 hidden nodes in CE and F respectively, dropout rate of 0.5, and mini-batch size of 64. In Table 5.3, it can be evaluated that

variant B seems more balanced in the recall and balanced accuracy rates. Where the recall rates in variant A are 0.4742, 0.5195 and 0.8486 for the *negative*, *neutral* and *positive* classes, variant B obtained recall rates with a relatively smaller gap in between the classes. This seems to indicate that variant B is more balanced in its performance. Specifically, variant B seems to predict the minority classes, i.e. *negative* and *neutral* rating classes, relatively better compared to variant A. The balanced accuracy (BA) rates validates this argument. As explained in Section 5.2.4, the BA rate drops to chance when the majority class overpowers the minority classes. Table 5.3 shows that the BA rates in variant A are relatively closer to randomness, i.e. $BA = 0.5$, than variant B. Hence, variant B obtained an average balanced accuracy of 0.7105 whereas variant A obtained an average balanced accuracy of 0.6634. Moreover, to select the best model, the objective function and its obtained values must be evaluated. Table 5.3 shows that variant A obtained loss values of 0.7477 and 0.9175 for the training and validation data; whereas variant B obtained a training loss of 0.7296 and validation loss of 0.8602. As addressed in Section 5.2, the purpose of the training procedure is to minimize the objective function. Even though the differences between variant A's and variant B's training loss are minimum, the gap between the training and validation loss for variant A is relatively larger with $\Delta = 0.1698$ than the difference in variant B with $\Delta = 0.1306$. The larger gap between training and validation loss could indicate that the training data does not provide sufficient information to learn the prediction task. On the basis of the balanced accuracy and result on the objective function, variant B is used in the remainder of the experiment.

	Variant A		**Variant B**	
	Recall	*Accuracy*	*Recall*	*Accuracy*
Negative	0.4742	0.6492	0.6758	0.7372
Neutral	0.5195	0.6689	0.5479	0.6775
Positive	0.8486	0.6722	0.8186	0.7167
Train Loss	0.7477		0.7296	
Validation Loss	0.9175		0.8602	

TABLE 5.3: Classification performances for selected baseline models

The second step is to apply the adversarial ensemble-based framework by training the debiased system. As discussed in Section 5.2.4, the debiased model uses the pretrained weights and architecture from baseline variant B for CE, ME and F. However, to mitigate the gender biases, the adversary A is integrated such that the adversarial minimax game can be used. Since baseline variant B contains 209 hidden nodes in F, the debiased model is initialized with 201 hidden nodes in A which is the average set of hidden nodes as explained in Section 5.2.3. All other parameter

setups are set as explained in Section 5.2 with the objective to minimize the categorical and binary cross-entropy cost function in F and A respectively. Even though the baseline model is trained by using early stopping, the debiased system is trained over 100 training steps without this method. This provided the model enough time to train and converge in the adversarial minimax game. Figure 5.6 shows the adversary (A) loss curves as training progresses during the adversarial game. It can be identified that A's loss was maximized and converged during training.

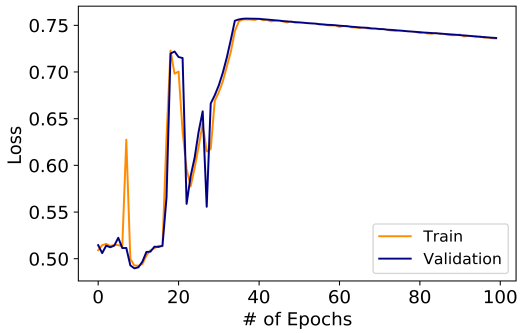

FIGURE 5.6: Adv. loss curves over epochs

	Recall	**Accuracy**
Negative	0.6532	0.7507
Neutral	0.4078	0.6363
Positive	0.8834	0.7099

TABLE 5.4: Debiased model performance

To validate whether A's ability to predict the gender z based on ME, Appendix C.2 shows the F and A prediction accuracy on validation set as training progresses. It was expected that the accuracy rate for A would stabilize around chance or $p = 0.5$ to ensure that ME is oblivious for the gender information z. But, it can be seen that the accuracy curves have stabilized around $p = 0.79$. Even though it seems that A can predict z, this is rather misleading and is not the case. Indeed, since 79 percent of the customers are males and 21 percent are females in the Amazon database, A's ability to predict z has achieved the level of chance. As (Brodersen et al., 2010) have argued, classification models can assign every observation to the majority class and achieve accuracy rates that mimics the imbalanced distribution. When assessing F's learning curve, Appendix C.2 shows that both the training and validation loss curve remains constant which is as expected. The weights for CE, ME and F were pretrained to obtain baseline variant B and has converged during the base training. Nevertheless, the training loss curve have reduced slightly where the validation loss increased with a relatively small difference. Thus, it can be expected that the debiased main task performance does not differ much. Appendix C.2 shows F's prediction accuracy over the number of training steps which indicates an increase in prediction performance. However, to verify whether the prediction performances are not reflecting the imbalanced database, Table 5.4 shows the recall and balanced accuracy rates. This indicates that the debiased system can be considered balanced in its performance. The recall rate is balanced to some extent where the balanced performance is validated by the balanced accuracy rates. Hence, the debiased model obtained an average balanced accuracy rate of 0.6990.

After training the baseline and debias models, the effectiveness of the adversarial ensemble-based method can be assessed. As mentioned in Section 2.3 and Section 5.2.4, the fairness notion focuses on ensuring that acceptable discrimination can occur in the predictive outcomes only through the dependence among admissible information. Therefore, the Positive Rate (PR) is used and derived for each gender group to assess whether the fairness constraint holds. Appendix D shows the confusion matrices for each product category and gender group. To examine the fairness notion, two product categories are selected to be assessed: (i) *Tools and Home* with Do-It-Yourself (DIY) and other home improvement products, and (ii) *Sport and Outdoors* with sports and outdoor products.

First, the *Tools and Home* category is evaluated. Table 5.5 shows the *PR*s for both the baseline and debiased systems. When evaluating the discrimination in the baseline performance, it can be identified that the model is biased to some extent. The *PR*s shows that male written reviews were more likely to be classified as *negative* and *neutral* rated products, whereas female written reviews were more likely to be predicted as *positive* rated products. The *PR*s for male and female groups across the rating classes are: $0.1268 > 0.1223$, $0.1609 < 0.1304$ and $0.7123 < 0.7473$. Based on these rates, it can be ascertained that the baseline system discriminates in its classification performance. Indeed, female reviewers are treated to be more positive in their predicted product ratings for Tools and Home products whereas male reviewers are considered to be more neutral en negative in their predicted ratings. Even though male written reviews are more likely to be classified as negative, the difference between the *PR*s for both gender groups in the negative class is minimum.

	Baseline			**Debiased**		
	Negative	*Neutral*	*Positive*	*Negative*	*Neutral*	*Positive*
Male	0.1268	0.1609	0.7123	0.1174	0.0998	0.7828
Female	0.1223	0.1304	0.7473	0.1170	0.0905	0.7925
Δ	0.0045	0.0305	0.0350	0.0004	0.0093	0.0097

TABLE 5.5: PR for *Tools and Home* category predictions for gender groups, with and without adversarial debiasing

When the debiasing performance is assessed, it can be seen that some *PR*s are reduced and approaches zero in the proportional differences across the gender groups. Indeed, the rates are approximately equal across male and female reviewers and product rating classes compared to the baseline model; from $\Delta = 0.0045$ to $\Delta = 0.0004$ in the *negative* class, $\Delta = 0.0305$ to $\Delta = 0.0093$ in the *neutral* class and $\Delta = 0.0350$ to $\Delta = 0.0097$ in the *positive* class. Thus, the adversarial debiasing

method seems to be effective in reducing the level of discrimination where the debiased system nearly satisfies the fairness constraint in the *Tools and Home* category.

The second product category to assess whether the fairness constraint holds is the *Sport and Outdoors* category. Table 5.6 shows the *PR*s for both the baseline and debiased systems. Based on the baseline performance, it can be identified that the system was biased due to some differences in the predicted proportions across the gender groups. The *PR* shows that female written reviews were more likely to be predicted as either *negative* or *positive* rated products whereas male written reviews were more likely to be classified as *neutral* rated products. The *PR*s for male and female groups across the product rating classes are: $0.0998 < 0.1039$, $0.1871 > 0.1690$ and $0.7131 < 0.7271$. Thus, it can be ascertained that the baseline model discriminates in the rating predictions for *Sports and Outdoor* products to some extent. The female reviewers are considered to be more critical and positive in their predicted product rating towards Sports and Outdoor products whereas male reviewers are treated to be more neutral in their predicted ratings. However, the bias seems to be less severe compared to the baseline system for Tools and Home products. Indeed, the *PR* differences across the gender groups are minimum with $\Delta = 0.0041$, $\Delta = 0.0181$ and $\Delta = 0.0140$ for the *negative*, *neutral* and *positive* classes.

	Baseline			**Debiased**		
	Negative	*Neutral*	*Positive*	*Negative*	*Neutral*	*Positive*
Male	0.0998	0.1871	0.7131	0.0886	0.1219	0.7895
Female	0.1039	0.1690	0.7271	0.1039	0.1126	0.7835
Δ	0.0041	0.0181	0.0140	0.0153	0.0093	0.0060

TABLE 5.6: PR for *Sports and Outdoors* category predictions for gender groups, with and without adversarial debiasing

After applying the debiasing method, it is expected that the differences in *PR*s between the proportion of each gender group becomes very close to zero since the differences were already relative small in the baseline setting. Indeed, for the debiased model, it can be assessed that the *PR*s are reduced and approaches zero in the proportional differences across the male and female written reviews for the predicted *neutral* and *positive* classes with $\Delta = 0.0093$ and $\Delta = 0.0060$. For the *negative* class, however, the difference in the *PR*s across the male and female reviewers increased from $\Delta = 0.0041$ in the baseline to $\Delta = 0.0153$ in the debiased model. This seems to be odd and was not expected because the adversarial method should mitigate gender information from the representation in *ME*. As seen in Table 5.6, the *PR*s between the baseline and debiased system with respect to female reviewers remains the same whereas the male's *PR* in the debiased model have decreased compared to

the *PR* in the baseline model. This pattern could be explained as a negative and unavoidable side-effect of the fairness notion as addressed in Section 2.3. Since the class distribution is imbalanced, the constraint can misclassify qualified examples on purpose to enforce equality in the *positive* class. Even though the increased differences across the gender groups in the *negative* rating class, the adversarial ensemble-based methods seems to be effective in reducing the gender bias in the model.

Category	**Baseline**	**Debiased**
All Beauty	0.1433	0.0627
Amazon Fashion	0.0647	0.0611
Arts and Crafts	0.0339	0.0304
Automotive	0.0257	0.0172
Scientific	0.0447	0.0382
Luxury Beauty	0.2194	0.1045
Musical Instruments	0.1233	0.1161
Video Games	0.0769	0.0432

TABLE 5.7: Average PR differences for all other product category predictions for gender groups, with and without adversarial debiasing

Based on these results, it can be determined that the adversarial debiasing method seems to be effective in mitigating the discrimination; where it nearly obeys the fairness constraint in both categories. However, to validate the consistency of the results, Table 5.7 shows the average *PR* differences across the gender groups and rating classes for all other product categories. It can be identified that the discrimination across the categories has reduced to some extent. For instance, the average *PR* differences between the baseline and debiased model have reduced significantly in the *All Beauty*, *Automotive*, *Scientific*, *Luxury Beauty* and *Video Games* categories. The average *PR* differences for the other categories, however, reduced with a smaller difference. This could be explained with the same side-effect as addressed for the discrepancy in the *Sports and Outdoors* category; where the constraint could misclassify qualified observations to balance the *PR*s across the target classes. Another explanation could be the number of examples per product category. Even though the test data contains 8.250 observations, the number of observations per product category could be too small. This could exaggerate or understate the differences across the categories. However, the adversarial debiasing method seems to be effective and nearly obeys the defined fairness constraint with only a small effect on the average balanced prediction accuracy in the overall model performance (71.05% vs. 69.90%).

6 Conclusion

Machine Learning (ML) has become an indispensable part of society. Due to the increasing amount and complexity of data, organizations are more reliant on ML to obtain insights and support decision-making. Among different ML methods, Deep Learning (DL) models have increased in industry usage. But, there is a growing concern that the use of DL can accentuate antisocial biases. Though, not all types of discrimination are unfair, this thesis proposed an adversarial ensemble-based DL framework such that acceptable discrimination can occur through the dependency on admissible information. To explain admissible discrimination, consider a Recommender System (RS). This system is viewed as fair when it treats gender groups with similar product interests similarly in the recommending process. But, when the groups have dissimilar interests, the discrimination in the recommended products across gender groups can be considered as reasonably fair. To create such a framework, three research questions were formulated which are summarized below.

The first research question was formulated as: *What adversarial debiasing approaches do exist and are effective to control and mitigate unintended biases in deep learning models?* In Section 3, two adversarial learning methods were examined and both were effective in their own setting. One method was focused on mitigating the protected information from the hidden representations by introducing an adversarial minimax game between two classifier components. The other approach achieved the same objective by considering the generated predictions in the adversarial game instead of the hidden representation. However, it was concluded that the latter method seemed to be more robust to obtain unbiased results. Since the weight update-rule is adjusted by adding a gradient projection term, the approach considered the possibility that the predictor and adversary can help each other during training. Based on this analysis, it was found that both methods were not able to control and mitigate biases at the same time. Both learning setups were designed to make the entire prediction model oblivious for the protected information.

The second research question was defined as: *How can the adversarial learning method be adjusted to debias hidden representations while allowing acceptable discrimination through admissible features?* In Section 4, an adversarial ensemble-based framework has been proposed which is defined as an ensemble of expert components. In this setting, it is proved that the adversarial minimax game optimum ensures that latent protected information can be mitigated from hidden representations in the Mitigation Expert while maintaining the information in the Control Expert. Also, it is shown

that a sample weighting scheme is required to control and decorrelate the relationship between admissible and protected features. To obtain more unbiased results, the adjusted weight update-rule with the gradient projection term is integrated in the framework. Based on these adjustments, it was proved that the acceptable discrimination can only occur through the dependency among the admissible features.

The third research question was defined as: *To what extent can the adversarial ensemble-based method control and mitigate unintended gender biases in product reviews such that rating predictions and gender information are conditional independent given the product categories?* In Section 5, a debiasing experiment was conducted on an Amazon product review database. This was used to validate the adversarial ensemble-based framework and theoretical claims. By evaluating the Positive Rate (PR) measure across a baseline and debias model, it is shown that the debiasing method indeed mitigates the gender biases and nearly obeys the fairness constraint. Based on the pre-selected *Tools and Home* and *Sports and Outdoor* product categories, it was ascertained that the baseline model was biased towards a gender group for each rating class in both contexts. After applying the adversarial debiasing method, the *PR* differences were reduced and approach zero across the gender groups for the *Tools and Home* category. For the *Sports and Outdoor* category, on the other hand, it was identified that the *PR* differences were reduced and approach zero in the *neutral* and *positive* rating classes. In the *negative* rating class, however, the *PR* difference increased. This performance discrepancy occurred to ensure equality in the *PR* across gender groups in the *positive* rating class which can be seen as a negative and unavoidable side-effect. Since the class distribution for each gender group is imbalanced, the constraint can misclassify qualified examples on purpose to balance the *PR*s across the classes. For all other product categories, it was found that the discrimination across the categories was reduced to some extent. While the average *PR* differences were reduced significantly for most categories, the average *PR* differences in the *Amazon Fashion*, *Arts and Crafts* and *Musical Instruments* reduced with a smaller difference. One explanation for this result was focused on the same negative side-effect that possibly occurred in the *Sports and Outdoor* category. Another explanation was that the number of examples per product category is not sufficient which could exaggerate or understate the *PR* differences. Even though the discrepancies and small proportional differences in the *PR*s, the adversarial ensemble-based method seems to be effective in reducing unintended gender biases with a small effect on the average balanced accuracy in the overall model performance.

To conclude, the adversarial ensemble-based framework is a reasonable method to control and mitigate unintended biases such that acceptable discrimination can only occur through admissible information. Policymakers and practitioners can add this method to their toolbox and use the approach when DL systems are used in business- as well as social-critical situations. In this work, one of the given example

cases in which the framework can be used was in review-based Recommender Systems (RS). When facing the rating sparsity problem, the adversarial ensemble-based method would ensure that customer reviews are classified in product rating classes without discriminating across gender groups. Thus, the rating similarities in the RS become more realistic and less biased such that gender groups that are interested in products from similar categories are treated similarly. Also, instead of mitigating the gender information from the entire model, allowing acceptable discrimination in the predictions only through product categories ensures that most predictive information is maintained without sacrificing the main task prediction performance.

As with any other research, this work contains some limitations and open question for future studies. First, the limitations are discussed. One limitation is the limited size of the database used in the experiment due to computational reasons. In the results, minimum differences in the average evaluation measures were identified for some product categories. This pattern was explained as a side-effect of the small number of observations per product category. More specifically, when product categories have little data, classifying observations differently could exaggerate or understate the actual debiasing effect. Thus, while the gender proportions per product category were maintained in the experiment, increasing the database and validate that each category has enough examples could solve the performance discrepancy.

A second limitation is that not all models have converged during training for which it was not possible to select the most optimal model. For instance, the selected baseline model that was used as starting point in the adversarial debiasing procedure still improved to some extent. Due to lack of computing resources, some concessions were made to reduce computational time in the training procedure such as the implementation of early stopping. However, when having enough computing resources, training the models over more train steps and use other training methods could validate the performances over the set of hyperparameter values.

The last limitation is the model architecture used. In the experiment, a simple Long Short-Term Memory (LSTM) model component is used to encode the text and protected data into a hidden representation while considering the word dependencies and sequential text order. But, there exists more advanced sequential architectures. For instance, extending the simple LSTM component to a bi-directional LSTM or even to the state-of-the-art Transformer with attention component could capture the stereotypical biases through the word dependencies in more detail. However, these methods are not used to validate the adversarial ensemble-based framework due to the required computational time and resources.

This thesis has some open questions that require future work to answer. One open question is: *does considering the protected attribute as an additional feature in the encoder indeed help the adversarial algorithm without sacrificing too much in the prediction*

accuracy? It was hypothesized that adding the protected attribute as an extra dimension into the contaminated embeddings help the debiasing method to be more effective in omitting the sensitive information from the hidden representation without reducing the quality of the generated prediction significantly. However, due to computational reasons, this hypothesis was not tested. Hence, the hypothesis can be validated by training the debiasing framework with and without considering the sensitive attribute as an extra dimension into the contaminated features.

The second open question is: *does the adversarial debiasing method remains effective while fine-tuning the pretrained GloVe embeddings or using other language models?* It was hypothesized that fine-tuning pretrained word vector would amplify and capture the gender biases in more detail. Since many studies have identified severe biases in all language models, fine-tuning the pretrained vectors would learn and reflect the discriminatory patterns for the prediction task at hand. Thus, to validate the effectiveness of the adversarial debiasing framework, the pretrained word embeddings should be fine-tuned and different language models should be used in different prediction task settings.

The last open question is: *is the adversarial debiasing method effective to satisfy the Equalized Odds fairness constraint?* This work focused on enforcing equal predictive rates such that the predictive outcomes and protected attribute are conditionally independent on the admissible information. This is fairness constraint is an equivalent of the Demographic Parity notion. To validate the robustness of the debiasing framework, the constraint could be extended by conditioning on the target class which enforces the notion of Equalized Odds. It is expected that the sample weighting scheme should be adjusted, but this must be proved and validated on its own.

Bibliography

Ali, M. et al. (Nov. 2018). "Discrimination through Optimization: How Facebook's Ad Delivery Can Lead to Biased Outcomes". In: *In Proceedings of the ACM on Human-Computer Interaction* 3. URL: https://doi.org/10.1145/3359301.

Allen, C. and T. Hospedales (2019). "Analogies Explained: Towards Understanding Word Embeddings". In: *Proceedings of the 36th International Conference on Machine Learning*. URL: arXiv:1901.09813.

Angwin, J. et al. (2016). "Machine Bias". In: *ProPublica*. URL: https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

Bengio, Y., P. Simard, and P. Frasconi (1994). "Learning long-term dependencies with gradient descent is difficult". In: *IEEE Transactions on Neural Networks*. URL: https://doi.org/10.1109/72.279181.

Beutel, A. et al. (2017). "Data decisions and theoretical implications when adversarially learning fair representations". In: URL: arXiv:1707.00075.

Binns, R. (2020). "On the apparent conflict between individual and group fairness". In: *FAT* '20: Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. URL: https://doi.org/10.1145/3351095.3372864.

Bolukbasi, T. et al. (2016). "Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings". In: *30th Conference on Neural Information Processing Systems (NIPS 2016)*. URL: arXiv:1607.06520v1.

Brodersen, K.H. et al. (2010). "The Balanced Accuracy and Its Posterior Distribution". In: *2010 20th International Conference on Pattern Recognition*. URL: https://doi.org/10.1109/ICPR.2010.764.

Caliskan, A., J.J. Bryson, and A. Narayanan (Apr. 2017). "Semantics derived automatically from language corpora contain human-like biases". In: *Science* 356, pp. 183–186. URL: https://doi.org/10.1126/science.aal4230.

Chen, J. et al. (2019). "Fairness Under Unawareness: Assessing Disparity When Protected Class Is Unobserved". In: *FAT* '19: Proceedings of the Conference on Fairness, Accountability, and Transparency*. URL: https://doi.org/10.1145/3287560.3287594.

Chen, L. et al. (Apr. 2018). "Investigating the Impact of Gender on Rank in Resume Search Engines". In: *CHI '18: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* 651, pp. 1–14. URL: https://doi.org/10.1145/3173574.3174225.

Chouldechova, A. (2017). "Fair prediction with disparate impact: A study of bias in recidivism prediction instruments". In: *FAT* '16: Proceedings of the Conference on Fairness, Accountability, and Transparency*. URL: arXiv:1703.00056.

Chouldechova, A. and A. Roth (2018). "The Frontiers of Fairness in Machine Learning". In: URL: arXiv:1810.08810.

Clark, C., M. Yatskar, and L. Zettlemoyer (2019). "Don't Take the Easy Way Out: Ensemble Based Methods for Avoiding Known Dataset Biases". In: *EMNLP 2019: 2019 Conference on Empirical Methods in Natural Language Processing*. URL: arXiv:1909.03683v1.

Courtland, R. (June 2018). "Bias detectives: the researchers striving to make algorithms fair". In: *Nature Research Journal*. URL: https://www.nature.com/articles/d41586-018-05469-3.

Dixon, L. et al. (2018). "Measuring and Mitigating Unintended Bias in Text Classification". In: *AIES '18: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. URL: https://doi.org/10.1145/3278721.3278729.

Du, M. et al. (2019). "Learning Credible Deep Neural Networks with Rationale Regularization". In: *ICDM 2019*. URL: arXiv:1908.05601.

Dwork, C. et al. (2012). "Fairness through awareness". In: *ITCS '12: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. URL: https://doi.org/10.1145/2090236.2090255.

Edizel, B. et al. (Mar. 2019). "FaiRecSys: mitigating algorithmic bias in recommender systems". In: *International Journal of Data Science and Analytics* 9, pp. 197–213. URL: https://doi.org/10.1007/s41060-019-00181-5.

Elazar, Y. and Y. Goldberg (2018). "Adversarial Removal of Demographic Attributes from Text Data". In: *EMNLP 2018: 2018 Conference on Empirical Methods in Natural Language Processing*. URL: arXiv:1808.06640v2.

Elman, J.L. (1990). "Finding Structure in Time". In: *Cognitive Science*. URL: https://doi.org/10.1207/s15516709cog1402_1.

Ganin, Y. and V. Lempitsky (2015). "Unsupervised domain adaptation by backpropagation". In: *ICML'15: Proceedings of the 32nd International Conference on International Conference on Machine Learning*. URL: https://dl.acm.org/doi/10.5555/3045118.3045244.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Goodfellow, I. et al. (2014). "Generative adversarial nets". In: *NIPS'14: Proceedings of the 27th International Conference on Neural Information Processing Systems*. URL: arXiv:1406.2661.

Hardt, M., E. Price, and N. Srebro (2016). "Equality of opportunity in supervised learning". In: *NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems*. URL: arXiv:1610.02413.

Hastie, T., R. Tibshirani, and J. Friedman (2017). *The Elements of Statistical Learning*. 2nd ed. Springer.

Hinton, G. (2012). "Neural Networks for Machine Learning". In: *Coursera, video lectures*.

Hinton, G.E. (Sept. 2002). "Training products of experts by minimizing contrastive divergence". In: *Neural Computation*. URL: https://doi.org/10.1162/089976602760128018.

Hochreiter, S. and J. Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Computation*. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

Jakob, N. et al. (Nov. 2009). "Beyond the stars: exploiting free-text user reviews to improve the accuracy of movie recommendations". In: *TSA '09: Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, pp. 57–64. URL: https://doi.org/10.1145/1651461.1651473.

Jozefowicz, R., W. Zaremba, and I. Sutskever (2015). "An empirical exploration of recurrent network architectures". In: *ICML'15: Proceedings of the 32nd International Conference on International Conference on Machine Learning*.

Kamiran, F. and T. Calders (2011). "Data preprocessing techniques for classification without discrimination". In: *Knowledge and Information Systems*. URL: https://doi.org/10.1007/s10115-011-0463-8.

Kearns, M. et al. (2018). "Preventing Fairness Gerrymandering: Auditing and Learning for Subgroup Fairness". In: *Proceedings of the 35th International Conference on Machine Learning*. URL: arXiv:1711.05144.

Keskar, N.S. et al. (2017). "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima". In: *International Conference for Learning Representations (ICRL) 2017*. URL: arXiv:1609.04836.

Kingma, D.P. and J. Ba (2015). "Adam: A Method for Stochastic Optimization". In: *3rd International Conference for Learning Representations (ICRL)*. URL: arXiv:1412.6980.

Kiritchenko, S. and S.M. Mohammad (2018). "Examining Gender and Race Bias in Two Hundred Sentiment Analysis Systems". In: *In Proceedings of the 7th Joint Conference on Lexical and Computational Semantics (*SEM), New Orleans, USA, 2018*. URL: arXiv:1805.04508v1.

Kleinberg, J., S. Mullainathan, and M. Raghavan (2016). "Inherent Trade-Offs in the Fair Determination of Risk Scores". In: *Proceedings of Innovations in Theoretical Computer Science (ITCS)*. URL: arXiv:1609.05807.

Kurita, K. et al. (2019). "Measuring Bias in Contextualized Word Representations". In: *1st ACL Workshop on Gender Bias for Natural Language Processing 2019*. URL: arXiv:1906.07337.

Liu, F. and B. Avci (2019). "Incorporating Priors with Feature Attribution on Text Classification". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. URL: https://doi.org/10.18653/v1/P19-1631.

Mansoury, M. et al. (2019). "Investigating Potential Factors Associated with Gender Discrimination in Collaborative Recommender Systems". In: *Association for the Advancement of Artificial Intelligence*. URL: arXiv:2002.07786v1.

Mikolov, T., W.T. Yih, and G. Zweig (2013). "Linguistic Regularities in Continuous Space Word Representations". In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. URL: https://www.aclweb.org/anthology/N13-1090.pdf.

Mikolov, T. et al. (2013). "Distributed representations of words and phrases and their compositionality". In: *NIPS'13: Proceedings of the 26th International Conference on Neural Information Processing Systems*. URL: arXiv:1310.4546.

Minaee, S. et al. (2020). "Deep Learning Based Text Classification: A Comprehensive Review". In: URL: arXiv:2004.03705.

Ni, J., J. Li, and J. McAuley (Nov. 2019). "Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 188–197. URL: https://doi.org/10.18653/v1/D19-1018.

Pascanu, R., T. Mikolov, and Y. Bengio (2013). "On the difficulty of training recurrent neural networks". In: *Proceedings of the 30 th International Conference on Machine Learning*. URL: arXiv:1211.5063.

Pennington, J., R. Socher, and C. Manning (2014). "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. URL: https://www.aclweb.org/anthology/D14-1162.pdf.

Prost, F., N. Thain, and T. Bolukbasi (2019). "Debiasing Embeddings for Reduced Gender Bias in Text Classification". In: *In Proceedings of the First Workshop on Gender Bias in Natural Language Processing (2019)*. URL: https://doi.org/10.18653/v1/W19-3810.

Rosenblatt, F. (1958). "The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain". In: *Psychological Review*, pp. 65–386.

Ross, A.S., M.C. Hughes, and F. Doshi-Velez (2017). "Right for the Right Reasons: Training Differentiable Models by Constraining their Explanations". In: *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. URL: https://doi.org/10.24963/ijcai.2017/371.

Ruder, S. (2016). "An overview of gradient descent optimization algorithms". In: URL: arXiv:1609.04747.

Rumelhart, D.E., G.E. Hinton, and R.J. Williams (Dec. 1986). "Learning representations by backpropagating errors". In: *Nature*. URL: https://doi.org/10.1038/323533a0.

Russell, S.J. and P. Norvig (2009). *Artificial Intelligence: A Modern Approach*. 3rd ed. Pearson.

Saeed, K. and V. Snášel (2014). "Computer Information Systems and Industrial Management". In: *13th IFIP TC 8 International Conference, CISIM 2014, Ho Chi Minh City, Vietnam, Proceedings*.

Sapiezynski, P. et al. (Dec. 2019). "Algorithms that "Don't See Color": Comparing Biases in Lookalike and Special Ad Audiences". In: *arXiv database*. URL: arXiv:1912.07579v2.

Seo, S. et al. (Sept. 2017). "Interpretable Convolutional Neural Networks with Dual Local and Global Attention for Review Rating Prediction". In: *RecSys '17: Proceedings of the Eleventh ACM Conference on Recommender Systems*, pp. 297–305. URL: https://doi.org/10.1145/3109859.3109890.

Soundarajan, S. and D.L. Clausen (2018). "Equal Protection Under the Algorithm: A Legal-Inspired Framework for Identifying Discrimination in Machine Learning". In: *Proceedings of the 35 th International Conference on Machine Learning*.

Sutskever, I., J. Martens, and G.E. Hinton (2011). "Generating Text with Recurrent Neural Networks". In: *Proceedings of the 28 th International Conference on Machine Learning*.

Tharwat, A. (2018). "Classification assessment methods". In: *Applied Computing and Informatics*. URL: https://doi.org/10.1016/j.aci.2018.08.003.

Xie, Q. et al. (2017). "Controllable Invariance through Adversarial Feature Learning". In: *31st Conference on Neural Information Processing Systems (NIPS 2017)*. URL: arXiv:1705.11122.

Zafar, M.B. et al. (2017). "Fairness Constraints: Mechanisms for Fair Classification". In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. URL: arXiv:1507.05259.

Zemel, R. et al. (2013). "Learning Fair Representations". In: *Proceedings of the 30th International Conference on Machine Learning*. URL: http://proceedings.mlr.press/v28/zemel13.pdf.

Zhang, B., B. Lemoine, and M. Mitchell (Dec. 2018). "Mitigating unwanted biases with adversarial learning". In: pp. 335–340. URL: arXiv:1801.07593v1.

Zhao, J. et al. (2017). "Men Also Like Shopping: Reducing Gender Bias Amplification using Corpus-level Constraints". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. URL: https://doi.org/10.18653/v1/D17-1323.

Zhao, J. et al. (June 2019). "Gender Bias in Contextualized Word Embeddings". In: *Proceedings of NAACL-HLT 2019*, pp. 629–634. URL: arXiv:1904.03310v1.

Zheng, L., V. Noroozi, and P.S. Yu (Feb. 2017). "Joint Deep Modeling of Users and Items Using Reviews for Recommendation". In: *WSDM '17: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 425–434. URL: https://doi.org/10.1145/3018661.3018665.

A Derivation of the Projected Gradients

Section 4.2 introduced the extended backpropagation weight update-rule. This is used to prevent gradients in the Mitigation Expert ME helping the adversary A reduce its loss. Moreover, the modification in ME's weight update-rule ensures that the effectiveness of the adversarial minimax game is maintained. Hence, the weights in ME are updated according to the following defined backpropagation rule:

$$\theta_{me} \leftarrow \theta_{me-1} - \eta(\frac{\partial L_f}{\partial \theta_{me-1}} - proj_{\frac{\partial L_a}{\partial \theta_{me-1}}} \frac{\partial L_f}{\partial \theta_{me-1}} - \lambda \frac{\partial L_a}{\partial \theta_{me-1}}) \tag{A.1}$$

In Section 4.2, all terms were defined except for the projection term $proj_{\frac{\partial L_a}{\partial \theta_{me-1}}} \frac{\partial L_f}{\partial \theta_{me-1}}$. This term requires some computational steps and is derived as follows:

$$proj_{\frac{\partial L_a}{\partial \theta_{me-1}}} \frac{\partial L_f}{\partial \theta_{me-1}} = \frac{x * u}{||u||^2} u \tag{A.2}$$

$$proj_{\frac{\partial L_a}{\partial \theta_{me-1}}} \frac{\partial L_f}{\partial \theta_{me-1}} = (x * u) u \tag{A.3}$$

$$||u|| = \frac{1}{||u||} u \tag{A.4}$$

Where $\frac{\partial L_a}{\partial \theta_{me-1}}$ is the partial derivative of weight matrix from ME w.r.t. the adversary's loss function L_a, and $\frac{\partial L_f}{\partial \theta_{me-1}}$ is the partial derivative of ME's weights w.r.t. the predictor's loss function L_f. To compute the projected gradients, the computation follows (A.2) in which x is the $\frac{\partial L_f}{\partial \theta_{me-1}}$ vector and u is a unit vector of $\frac{\partial L_a}{\partial \theta_{me-1}}$. However, this expression can be simplified into (A.3) by considering $||u|| = 1$. To ensure that $||u|| = 1$, $||u||$ must be normalized by (A.4). As a result, the gradients in ME are prevented from moving in a direction that helps A decrease its loss by orthogonally projecting the $\frac{\partial L_f}{\partial \theta_{me-1}}$ vector onto $\frac{\partial L_a}{\partial \theta_{me-1}}$.

B Amazon Product Review Database

B.1 Gender Distribution per Product Category (in %)

Product Category	Female	Male
All Beauty	73.5	26.5
Amazon Fashion	83.1	16.9
Appliances	31.0	69.0
Arts and Crafts	80.6	19.4
Automotive	9.0	91.0
Cell Phones and Accessories	36.1	63.9
Digital Music	45.3	54.7
Gift Cards	59.2	40.8
Grocery and Gourmet Food	58.5	41.5
Scientific	11.6	88.4
Luxury Beauty	75.2	24.8
Magazine Subscriptions	57.9	42.1
Musical Instruments	10.8	89.2
Office Products	48.7	51.3
Patio, Lawn and Garden	32.3	67.7
Prime Pantry	63.9	36.1
Software	20.9	79.1
Sports	20.9	79.1
Tools and Home	17.1	82.9
Video Games	20.6	79.4

C Model Training Performances

C.1 Baseline Learning Curves

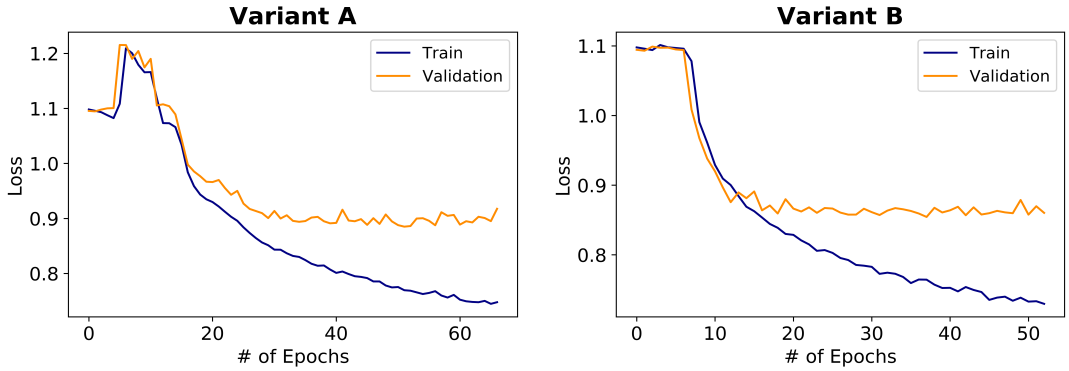

FIGURE C.1: The loss learning curve for Variant A and Variant B

C.2 Debias Learning Curves

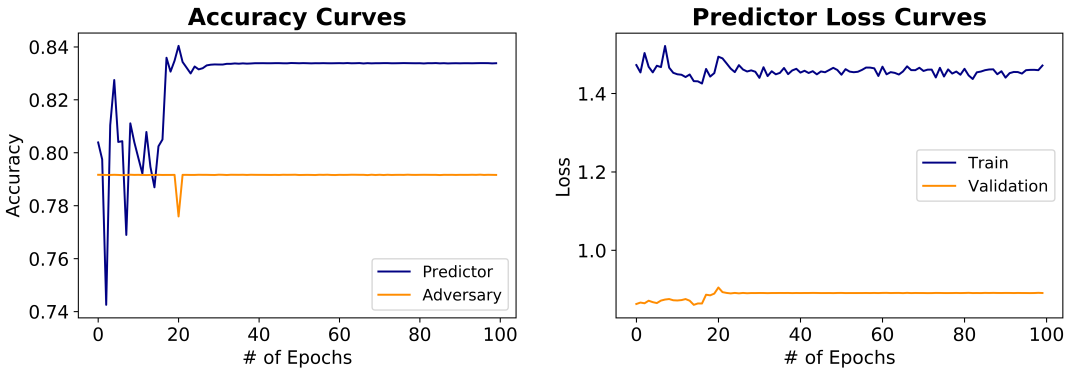

FIGURE C.2: Conventional accuracy curves on validation data (left) and Predictor loss curve on train and validation data (right)

D Debiasing Evaluation

D.1 Product Category *Tools and Home Improvement*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	89	32	6	127
	Neutral	33	60	22	115
	Positive	94	182	1,185	1,461
	Total	216	274	1,213	1,703

TABLE D.1: Confusion matrix for Baseline model on *Tools and Home* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	22	4	2	28
	Neutral	7	11	6	24
	Positive	17	34	273	324
	Total	46	49	281	376

TABLE D.2: Confusion matrix for Baseline model on *Tools and Home* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	93	21	13	127
	Neutral	30	45	40	115
	Positive	77	104	1,280	1,461
	Total	200	170	1,333	1,703

TABLE D.3: Confusion matrix for Debiased model on *Tools and Home* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	20	5	3	28
	Neutral	7	10	7	24
	Positive	17	19	288	324
	Total	44	34	298	376

TABLE D.4: Confusion matrix for Debiased model on *Tools and Home* category predictions and *Female* customers

D.2 Product Category *Sports and Outdoors*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	100	60	10	170
	Neutral	41	99	35	175
	Positive	90	274	1,605	1,969
	Total	231	433	1,650	2,314

TABLE D.5: Confusion matrix for Baseline model on *Sports and Outdoors* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	30	11	3	44
	Neutral	12	21	11	44
	Positive	17	64	399	480
	Total	59	96	413	568

TABLE D.6: Confusion matrix for Baseline model on *Sports and Outdoors* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	94	54	22	170
	Neutral	37	72	66	175
	Positive	74	156	1,739	1,969
	Total	205	282	1,827	2,314

TABLE D.7: Confusion matrix for Debiased model on *Sports and Outdoors* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	32	7	5	44
	Neutral	13	17	14	44
	Positive	14	40	426	480
	Total	59	64	445	568

TABLE D.8: Confusion matrix for Debiased model on *Sports and Outdoors* category predictions and *Female* customers

D.3 Product Category *All Beauty*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	10	4	3	17
	Neutral	21	47	16	84
	Positive	1	2	7	10
	Total	32	53	26	111

TABLE D.9: Confusion matrix for Baseline model on *All Beauty* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	75	22	8	105
	Neutral	8	39	12	59
	Positive	9	18	108	135
	Total	92	79	128	299

TABLE D.10: Confusion matrix for Baseline model on *All Beauty* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	13	3	1	17
	Neutral	22	38	24	84
	Positive	0	2	8	10
	Total	35	43	33	111

TABLE D.11: Confusion matrix for Debiased model on *All Beauty* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	76	24	5	105
	Neutral	4	42	13	59
	Positive	12	24	99	135
	Total	92	90	117	299

TABLE D.12: Confusion matrix for Debiased model on *All Beauty* category predictions and *Female* customers

D.4 Product Category *Amazon Fashion*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	13	6	2	21
	Neutral	1	3	1	5
	Positive	3	5	29	37
	Total	17	14	32	63

TABLE D.13: Confusion matrix for Baseline model on *Amazon Fashion* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	28	12	7	47
	Neutral	9	32	8	49
	Positive	10	27	139	176
	Total	47	71	154	272

TABLE D.14: Confusion matrix for Baseline model on *Amazon Fashion* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	13	7	1	21
	Neutral	0	4	1	5
	Positive	4	7	26	37
	Total	17	18	29	63

TABLE D.15: Confusion matrix for Debiased model on *Amazon Fashion* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	29	11	7	47
	Neutral	13	28	8	49
	Positive	10	33	133	176
	Total	52	72	148	272

TABLE D.16: Confusion matrix for Debiased model on *Amazon Fashion* category predictions and *Female* customers

D.5 Product Category *Arts and Crafts*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	2	1	0	3
	Neutral	0	5	2	7
	Positive	3	7	96	106
	Total	5	13	98	116

TABLE D.17: Confusion matrix for Baseline model on *Arts and Crafts* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	9	6	1	16
	Neutral	4	12	3	19
	Positive	9	43	316	368
	Total	22	61	320	403

TABLE D.18: Confusion matrix for Baseline model on *Arts and Crafts* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	2	1	0	3
	Neutral	0	3	4	7
	Positive	3	5	98	106
	Total	5	9	102	116

TABLE D.19: Confusion matrix for Debiased model on *Arts and Crafts* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	8	5	3	16
	Neutral	5	10	4	19
	Positive	12	27	329	368
	Total	25	42	336	403

TABLE D.20: Confusion matrix for Debiased model on *Arts and Crafts* category predictions and *Female* customers

D.6 Product Category *Automotive*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	60	16	4	80
	Neutral	15	32	8	55
	Positive	58	94	738	890
	Total	133	142	750	1,025

TABLE D.21: Confusion matrix for Baseline model on *Automotive* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	10	1	1	12
	Neutral	3	7	4	14
	Positive	4	9	70	83
	Total	17	17	75	109

TABLE D.22: Confusion matrix for Baseline model on *Automotive* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	56	20	4	80
	Neutral	14	26	14	55
	Positive	44	67	779	890
	Total	114	113	797	1,025

TABLE D.23: Confusion matrix for Debiased model on *Automotive* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	9	2	1	12
	Neutral	4	7	3	14
	Positive	1	4	78	83
	Total	14	13	82	109

TABLE D.24: Confusion matrix for Debiased model on *Automotive* category predictions and *Female* customers

D.7 Product Category *Scientific*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	0	1	0	1
	Neutral	2	1	1	4
	Positive	1	7	55	63
	Total	3	9	56	68

TABLE D.25: Confusion matrix for Baseline model on *Scientific* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	0	0	0	0
	Neutral	0	0	1	1
	Positive	1	1	6	8
	Total	1	1	7	9

TABLE D.26: Confusion matrix for Baseline model on *Scientific* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	1	0	0	1
	Neutral	1	3	0	4
	Positive	7	7	49	63
	Total	9	10	49	68

TABLE D.27: Confusion matrix for Debiased model on *Scientific* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	0	0	0	0
	Neutral	0	0	1	1
	Positive	1	1	6	8
	Total	1	1	7	9

TABLE D.28: Confusion matrix for Debiased model on *Scientific* category predictions and *Female* customers

D.8 Product Category *Luxury Beauty*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	0	0	0	0
	Neutral	0	1	1	2
	Positive	0	0	11	11
	Total	0	1	12	13

TABLE D.29: Confusion matrix for Baseline model on *Luxury Beauty* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	0	1	0	1
	Neutral	0	0	3	3
	Positive	0	7	16	23
	Total	0	8	19	27

TABLE D.30: Confusion matrix for Baseline model on *Luxury Beauty* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	0	0	0	0
	Neutral	1	0	1	2
	Positive	0	2	9	11
	Total	1	2	10	13

TABLE D.31: Confusion matrix for Debiased model on *Luxury Beauty* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	0	0	1	1
	Neutral	0	0	3	3
	Positive	1	1	21	23
	Total	1	1	25	27

TABLE D.32: Confusion matrix for Debiased model on *Luxury Beauty* category predictions and *Female* customers

D.9 Product Category *Musical Instruments*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	12	5	0	17
	Neutral	2	8	4	14
	Positive	11	22	149	182
	Total	25	35	153	213

TABLE D.33: Confusion matrix for Baseline model on *Musical Instruments* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	2	4	0	6
	Neutral	2	2	0	4
	Positive	0	4	16	20
	Total	4	10	16	30

TABLE D.34: Confusion matrix for Baseline model on *Musical Instruments* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	11	5	1	17
	Neutral	3	5	6	14
	Positive	7	10	165	182
	Total	21	20	172	213

TABLE D.35: Confusion matrix for Debiased model on *Musical Instruments* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	2	4	0	6
	Neutral	2	2	0	4
	Positive	0	1	19	20
	Total	4	7	19	30

TABLE D.36: Confusion matrix for Debiased model on *Musical Instruments* category predictions and *Female* customers

D.10 Product Category *Video Games*

Baseline		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	32	14	1	47
	Neutral	10	30	6	46
	Positive	27	75	241	343
	Total	69	119	248	436

TABLE D.37: Confusion matrix for Baseline model on *Video Games* category predictions and *Male* customers

Baseline		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	5	3	0	8
	Neutral	3	1	1	5
	Positive	1	17	64	82
	Total	9	21	65	95

TABLE D.38: Confusion matrix for Baseline model on *Video Games* category predictions and *Female* customers

Debiased		**Predicted**			
Male		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	30	14	3	47
	Neutral	11	20	15	46
	Positive	24	33	286	343
	Total	65	67	304	436

TABLE D.39: Confusion matrix for Debiased model on *Video Games* category predictions and *Male* customers

Debiased		**Predicted**			
Female		*Negative*	*Neutral*	*Positive*	*Total*
Actual	*Negative*	5	2	1	8
	Neutral	2	1	2	5
	Positive	1	14	67	82
	Total	8	17	70	95

TABLE D.40: Confusion matrix for Debiased model on *Video Games* category predictions and *Female* customers