Fairness without Demographics through Adversarially Reweighted Learning

E4040.2021Fall.FAIR.report
Riya Gupta rg3332, Kate Harwood krh2154, Andrea Lin al4213
Columbia University

Abstract

This paper aims to apply the concept of Adversarial Reweighted Learning (ARL) as a tool to improve the fairness of a Machine Learning (ML) model when protected features such as race or gender are not known or cannot be used. In fairness research, these protected features are used to produce fairer models, however in the "real world" this data is sometimes prohibited from use, or not available. This paper strives to produce a method that is more realistically helpful for producing fair results in the "real world" by creating a fair model without the use of protected features. In our project, we reproduce the models using the same methodology and attempt to reproduce the paper's results. Our results, while following the same general trend of the paper's, do not show some of the same improvements in metrics for protected groups as the paper produces.

1. Introduction

The motivation for this research is to address fairness concerns within ML systems. Recent research has displayed accuracy disparities across demographic groups for tasks such as facial detection, health-care systems, and recommendation systems [1, 2, 3]. These disparities can have repercussions in decision making. Therefore, many researchers have focused on proposing "debiasing" methods in order to improve ML fairness among individuals [7, 8, 9, 10] and among known groups [2, 4, 5]. However, these current "debiasing" methods are flawed: they rely on the knowledge of protected group status in order to produce unbiased models.

This project replicates a paper that presents a unique solution to the challenge of improving predictive fairness between protected groups *when knowledge of protected group status is unknown*. In many instances, with privacy, legal, or regulatory restrictions, protected features such as race, gender and ethnicity cannot be collected or used in ML models. Models without protected features as input still produce the same disparities across protected groups as those that do include protected features, but it is even more difficult to "debias" these models.

The authors of the paper measure fairness using Rawlsian Max-Min Fairness which maximizes the minimum utility across protected groups, allowing for inequalities between groups [6]. A related measure of fairness worth noting used in many other research papers [4, 5, 11, 12, 13] is parity based fairness, which only takes into consideration gaps across groups. While parity-based fairness aims to decrease the statistical gap between groups, Rawlsian Max-Min fairness aims to improve the performance of worst-off protected groups. The two notions are similar, but Rawlsian Max-Min fairness directly decreases the bias seen in worse-off protected groups, rather than allowing for a decrease in the accuracy of better performing groups.

This paper presents a novel approach to improve fairness when protected group membership is unknown through Adversarial Reweighted Learning (ARL), a modeling approach that aims to improve the utility for worst-off protected groups, without access to protected features at training or inference time.

In this paper, ARL exploits "computationally-identifiable errors" through an Adversary, in order to improve the performance of worst-off groups. The most similar work to this paper is [14], which utilizes distributionally robust optimization (DRO) to achieve Rawlsian Max-Min fairness when protected group membership is unknown. However, the two methods differ in how they identify protected groups: DRO relies on outliers in the classification task, and the authors of this paper use a concept they call "computational-identifiability" to pinpoint protected groups (more on this in Methodology).

2. Summary of the Original Paper

2.1 Methodology of the Original Paper

The architecture consists of a standard feed-forward network to implement both Learner and Adversary. The Learner is a fully connected two layer feed-forward network with 64 and 32 hidden units in the hidden layers, with ReLU activation function. The Adversary assigns weights to the examples based on the Learner's example loss. Both models are trained alternately, and the main loss used is binary cross entropy loss. The loss equation (using the notion of Rawlsian Min-Max Fairness) is :

$$J(\theta, \phi) = \min_{\theta} \max_{\phi} \sum_{i=1}^{n} \lambda_{\phi}(x_i, y_i) \cdot \ell_{ce}(h_{\theta}(x_i), y_i)$$

The inputs to the Primary Learner are the non-protected features from the dataset (X), and the inputs to the Adversary are the non-protected features and the labels (X and Y). Figure 1 shows the architecture used in the paper.

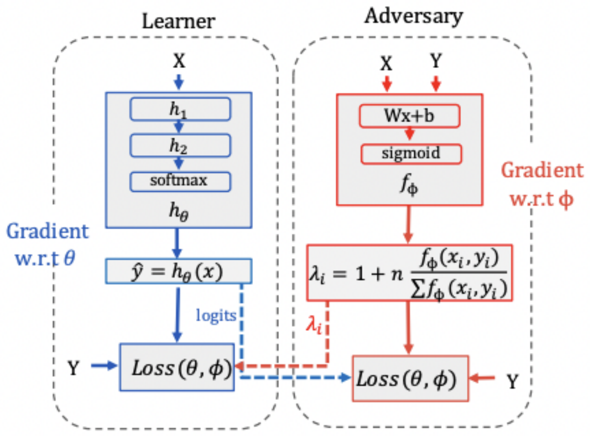


Figure 1. Model Architecture

It is worth noting the adversarial weights **λ** in this figure**.** These weights are more salient for computationally-identifiable groups, putting more emphasis on the examples for which the loss is the highest in the Learner.

Data

The original paper uses 3 datasets.

1. Adult
2. LSAC
3. COMPAS

The **Adult** dataset has a sample size of 40,701 and 15 features with the prediction task of determining whether a person makes income over $50,000 and originated from the United States Census Bureau. The **LSAC** dataset has a sample size of 27,479 and 12 features with the prediction task of determining if an individual passed the bar exam. This data was collected in response to reports that suggested lower passing rates in minority groups. The **COMPAS** dataset has a sample size of 7,215 and 11 features with a prediction task of determining whether an individual would recidivate in two years. The data was collected by Northpointe Inc. to determine the validity of their recidivism algorithm and discover patterns within groups.

Table 1 shows a description of the datasets used in the original paper. The protected features for all datasets are race and sex. We use the LSAC and COMPAS datasets for this project.

Dataset	Size	No. features	Protected features	Protected groups	Prediction task
Adult	40701	15	Race, Sex	{White, Black} $\times$ {Male, Female}	income $\geq$ 50k?
LSAC	27479	12	Race, Sex	{White, Black} $\times$ {Male, Female}	Pass bar exam?
COMPAS	7215	11	Race, Sex	{White, Black} $\times$ {Male, Female}	recidivate in 2 years?

Table 1: Description of the Datasets

Computational-Identifiability

The key difference in this paper's approach to adversarial reweighted learning from the approaches in the previous literature is in the notion of computationally-identifiable subgroups. In [14], the DRO version of the task also used reweighted learning to improve results for subgroups whose membership is not known, however the authors of this current paper improved on the results by tweaking the implementation of the reweighting. The paper using the DRO method [14] simply upweighted the examples with the worst training error, the outliers (those in red in Figure 2). The authors of this paper instead rely on the notion of computationally-identifiable errors. Instead of focusing on improving results for potentially noisy outliers, they focus on improving results for a group of misclassified examples (for example, the circled green plusses in Figure 2). Their hypothesis was that focusing on these computationally-identifiable misclassified regions of protected subgroups would increase the AUC for those protected subgroups. They use the Adversary to leverage this notion.

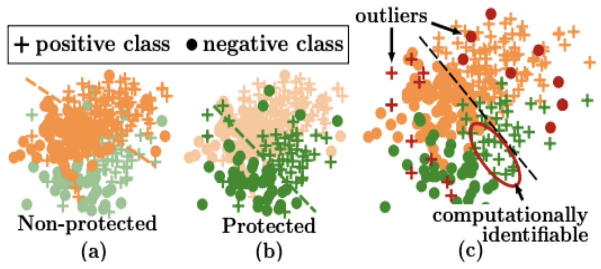


Figure 2. Computational-Identifiability

Hyperparameters

Each dataset for the paper is randomly split into 70% training and 30% test sets. 5-fold cross validation was used to pick model hyperparameters, including batch size

(32, 64, 128, 256, 512) and learning rate (0.001, 0.01, 0.1, 1, 2, 5). The final hyperparameters were batch size of 32 and learning rate of 0.001.

Evaluation Metrics

AUC (area under the ROC curve) is used as the main utility metric. ROC (receiver operating characteristics curve) is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. One of the reasons the authors chose AUC as their main metric is because of its robustness to class imbalance: some of the datasets in this research are highly imbalanced. It is also agnostic to the decision threshold in a classification task and contains both *false negative rate (FNR)* and *false positive rate (FPR)*

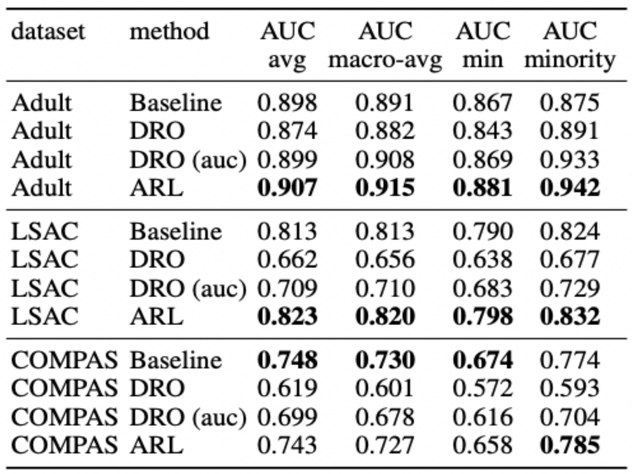
The paper reports 4 different AUCs:

1. AUC (avg): average AUC over all groups, protected and non-protected
2. AUC (min): minimum AUC over all protected groups (score for the protected group that had the smallest AUC)
3. AUC (macro-avg): macro-average over all protected group AUCs.
4. AUC (minority): AUC reported for the smallest protected group in the dataset.

2.2 Key Results of the Original Paper

Along with a Baseline and ARL model, the authors also compare their results to an Inverse Probability Weighting (IPW) model (not shown) and the DRO model from previous literature. Table 2 shows the original experimental results of the paper. The authors ran their models multiple times; the results shown are the average AUC metrics for each model.

The authors find that the ARL model has a higher AUC performance than the Baseline and DRO models for the Adult and LSAC datasets. They mention that the COMPAS dataset is known to have noisy target labels, and posit this as a reason for their failure to show improvement over the Baseline on this dataset.



dataset	method	AUC avg	AUC macro-avg	AUC min	AUC minority
Adult	Baseline	0.898	0.891	0.867	0.875
Adult	DRO	0.874	0.882	0.843	0.891
Adult	DRO (auc)	0.899	0.908	0.869	0.933
Adult	ARL	**0.907**	**0.915**	**0.881**	**0.942**
LSAC	Baseline	0.813	0.813	0.790	0.824
LSAC	DRO	0.662	0.656	0.638	0.677
LSAC	DRO (auc)	0.709	0.710	0.683	0.729
LSAC	ARL	**0.823**	**0.820**	**0.798**	**0.832**
COMPAS	Baseline	**0.748**	**0.730**	**0.674**	0.774
COMPAS	DRO	0.619	0.601	0.572	0.593
COMPAS	DRO (auc)	0.699	0.678	0.616	0.704
COMPAS	ARL	0.743	0.727	0.658	**0.785**

Table 2. Main Results

3. Methodology (of the Students' Project)

Our project aims to reproduce the results of the ARL model. We use the same model architecture as the paper and the same hyperparameters (with a change to the learning rate for one dataset). We focus on reproducing results for the LSAC and COMPAS datasets for the Baseline and ARL model. The networks are constructed with TensorFlow 2.0 and Keras.

3.1. Objectives and Technical Challenges

Broadly, we aim to:

- Process the raw datasets available online
- Implement TensorFlow 2.0 and Keras version of the original paper's Baseline and ARL Model
- Use the AUC metric to assess our results and compare them to those in the original paper

Challenges

Data Parsing : One of the biggest challenges was parsing the raw data. Due to the lack of clean data, it took some effort to pull and parse the real-world data. The latter part of this is shown in our "data_utils" folder code, but even getting the data into a CSV datatable that our code could parse to generate input features took some work. Additionally, the datasets are small after cleaning and preprocessing, which can make it hard for the models to learn meaningful relationships.

Feature Generation : Much of the data is made up of categorical features instead of the numerical features. Protected features like "sex" and "race" and unprotected features like "pass_bar" or "is_recid" can be binarized into 0 and 1, however, features like “c_charge_desc” cannot be binarized and have multiple values. We tried various methods like one-hot encoding and word embeddings to extract these features.

Model and Training : This project required us to develop a novel adversarial architecture in Keras with few online resources.

Complexity: Frequently in deep learning, the complexity of a project lies in its large model architecture, vast amount of training data and hours of training time required. The intricacies of this project do not reside in the complexity of model architecture or amount of training data, but rather in the problem it is trying to solve and the methods it uses. This is especially true of the role of the Adversary and the idea of computational-identifiability, and of the added complexity of working with protected and unprotected features. Understanding approaches to debiasing, and why some methods may work better than others in certain scenarios is a complex topic. While many models are built simply to produce higher numbers on base metrics, models built to be fair attempt to maximize these numbers in a thoughtful and nuanced way.

3.2. Problem Formulation and Design Description

In this section, we show the workflow of our problem formulation and network design. We follow the basic deep learning training pipeline which includes data preprocessing, model generation, training and testing processes.

The raw dataset is parsed and prepared such that it is compatible for processing in Keras. In the next step, the models are created and trained with appropriate hyperparameters and loss functions. The test data is split into groups based on protected features and quantitative results are obtained for these protected groups. Our results are compared with the original paper’s results. Figure 3 shows the visual representation of the workflow.

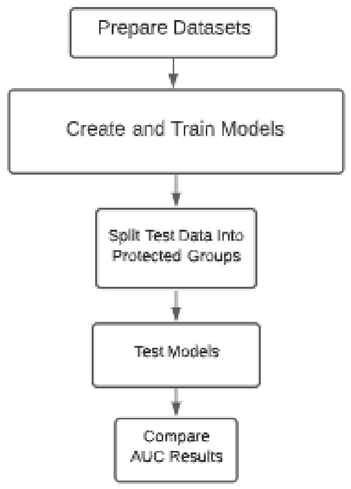


Figure 3. Problem Formulation FlowChart

4. Implementation
4.1 Data

We used the LSAC and COMPAS datasets available online. This is raw data, and it required quite a bit of parsing to transform it into a format that can be read in by a Keras model. Some of the features are natural language categorical features, and the original paper uses embeddings to encode them. We experimented with different ways of processing these features, and tried training the models on different combinations. Our final COMPAS dataset leaves out of one of the natural language categorial features. The LSAC dataset we leave intact. We split the data into train and test sets (opting to use all the available data for training and reliable test results rather than include eval data, since the datasets are small).

Figure 4 and Figure 5 present an overall distribution count of (sex,race) in COMPAS and LSAT dataset respectively, which gives us an insight into the distribution of the overall data in terms of protected groups, and shows the data imbalance when stratifying by these groups.

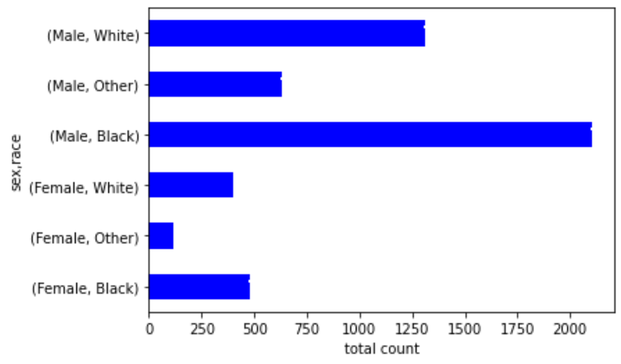


Figure 4. Data distribution of COMPAS

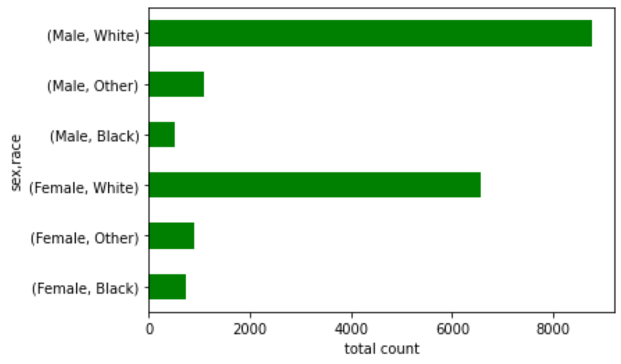


Figure 5. Data distribution of LSAC

4.2 Deep Learning Network

Figure 6 shows the architecture of the Primary and Adversary model. The Primary model begins with input (X) of non-protected features. These features go through two dense layers with 64 neurons and 32 neurons respectively and ReLU activation, then a final dense layer with 1 neuron and sigmoid activation. Raw predictions are made from the layers and then used to calculate the loss of both the Primary and Adversary model. The loss calculation for the Primary model also requires the class label (Y) and the rescaled weights λ from the Adversary model.

The Adversary model takes inputs (X) of non-protected features and Y of class labels. These inputs go through a dense layer of 32 neurons with ReLU activation and then a dense layer of 1 neuron with sigmoid activation. The rescaled weights are then used to calculate the loss of both the Primary and the Adversary model. The loss calculation for the Adversary model also requires the raw predictions obtained from the Primary model and class label (Y).

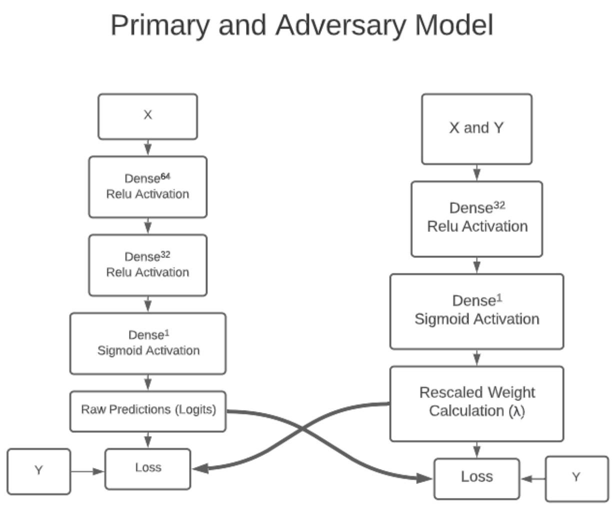


Figure 6. Primary and Adversary Model

Figure 7 shows the architecture of the Baseline model. This model is created in order to analyze the changes in the results made by including an Adversary with the Primary model. Similar to the Primary model, it takes input (X) of non-protected features and passes the input through a dense layer of 64 neurons with ReLU activation. For a fair comparison with the Primary-Adversary model, the Baseline has 64 neurons in its second layer (rather than 32) to account for the extra capacity of the ARL model with the Adversary's 32 neurons. As with the Primary and Adversary models, data is then fed through a final dense layer of 1 neuron with sigmoid activation where the raw predictions are calculated. The loss is then calculated using the raw predictions and class label (Y).

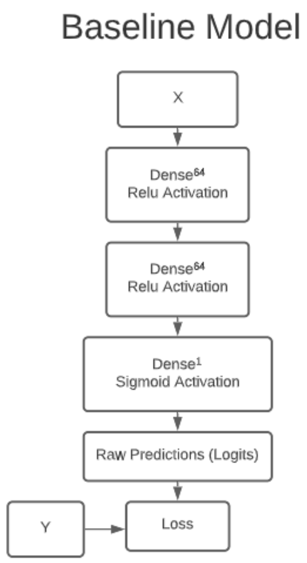


Figure 7. Baseline Model

4.3 Code Architecture

The full schematic of the code architecture used in this project is shown in Figure 8. It can be broken down into steps of preparing the data, creating and training the models, testing the models, and analyzing the results.

In the data preparation phase, the raw data from COMPAS and LSAC is first parsed into CSV files. Afterwards, the data is parsed into features that are compatible with our Keras and TensorFlow 2.0 based models.

Next, the models are created and trained. The same schematic can be applied to the Baseline model used for comparison. Once the models are trained, the test data is split into protected groups to compute the various AUC metrics. Finally, the results are compared to the results from the original paper's results.

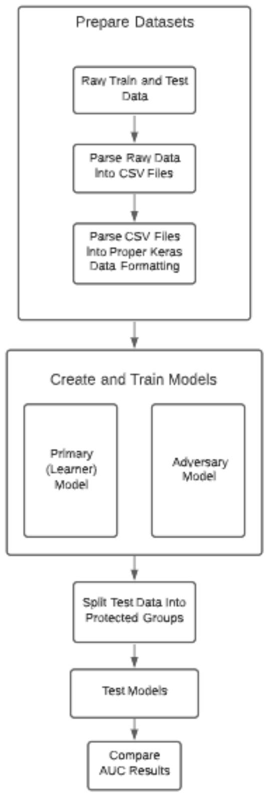


Figure 8. Code Architecture

5. Results

In this section, we demonstrate our key findings and results from our models.

5.1 Project Results

Train Time: Data parsing and generation for the datasets takes about ~ 3 minutes each. Training time is ~ 5 minutes for each model (Baseline and ARL). Models trained on a Macbook Air with M1 chip and 16GB memory.

Results: Our final metrics are the result of an average of 5 training runs for each model. We found that sometimes the Baseline produces better results and sometimes ARL produces better results, but on average, the Baseline model outperforms the ARL model on all metrics.

We made several attempts to increase metrics, including:

- making slight changes to the implementation of the loss functions of the models
- trying Hinge loss instead of Binary Cross Entropy loss in the adversary model (the original paper also mentioned experimenting with this)
- tuning hyperparameters, including trying different learning rates and hidden layer sizes
- pruning the data so that it is balanced and running models on the altered data
- using batched vs. non-batched data

While most of the tweaks had minimal effect, we found that, for the COMPAS dataset, a learning rate of 0.01 produced better overall results with both models than the paper's learning rate of 0.001.

Table 3 shows our results on the metrics in the original paper, and Table 4 shows our results on extra metrics including overall Accuracy and AUC for the largest protected group. Table 5 shows the difference in results between the Baseline and ARL models. The gap in scores is smaller on the COMPAS dataset than on the LSAC dataset.

dataset	**method**	**AUC avg**	**AUC macro_avg**	**AUC min**	**AUC minority**
LSAC	Baseline	**0.784**	**0.749**	**0.790**	**0.713**
LSAC	ARL	0.722	0.703	0.730	0.687
COMPAS	Baseline	**0.678**	**0.659**	**0.679**	**0.633**
COMPAS	ARL	0.666	0.643	0.662	0.629

Table 3: Our results

dataset	method	accuracy	AUC largest protected group
LSAC	Baseline	**0.810**	**0.784**
LSAC	ARL	0.800	0.719
COMPAS	Baseline	**0.656**	**0.684**
COMPAS	ARL	0.602	0.658

Table 4: Extra Results

dataset	AUC avg: difference	AUC macro_avg: difference	AUC min: difference	AUC minority: difference
LSAC	0.061	0.046	0.060	0.027
COMPAS	0.011	0.015	0.016	0.004

Table 5: Our results: Difference between Baseline and ARL models

5.2 Comparison of the Results Between the Original Paper and Students' Project

dataset	method	AUC avg: diff	AUC macro_avg: diff	AUC min: diff	AUC minority: diff
LSAC	Baseline	0.029	0.064	0	0.111
LSAC	ARL	0.101	0.117	0.068	0.145
COMPAS	Baseline	0.070	0.131	0.005	0.141
COMPAS	ARL	0.077	0.155	0.004	0.156

Table 6: Difference between the original paper's results and our results

Though our results are mostly lower than the results in the paper, and we fail to reproduce the increase in AUC metrics on the LSAC data that the paper shows, we observe some similar trends in differences between the AUC metrics in general, on both datasets and models.

Table 6 shows the differences in the AUC scores between the paper's results and our results. The difference in our ARL metrics and the paper's ARL metrics is similar to the differences between our Baseline results and their Baseline results, which may indicate that the failure to reproduce the paper's results is not a failure in our ARL architecture but rather due to outside factors that impacted the overall results of our experiment. These potential factors are discussed further below.

5.3 Discussion of Insights Gained

One large difference between our implementation and the paper's implementation is that the authors of this paper used TensorFlow 1.0, and used tf Estimators instead of Keras. This has the potential to change the results, even if the data, model architecture and hyperparameters are the same. For example, though unlikely, it is possible that Keras uses a slightly different implementation of AUC than TensorFlow 1.0 used, so even if our models are of similar quality, the final metrics could be skewed. There are many places where implementation differences in built-in functions could cause different results throughout the entire pipeline of data collection to results.

We also do not know certain hyperparameters for the models in the paper, like exactly how long they trained their final models for, final batch size used, etc., which could account for some of the differences in the results we see. They also mention that the COMPAS dataset is imbalanced, but do not mention the LSAC dataset is, although we have noticed that it is highly unbalanced (most of the examples are "not-pass"). This could imply the data changed in some way or they were using a different version than we are.

Overall, we attempted to reconstruct the models and data as faithfully as possible to the original paper, but there are always bound to be discrepancies in two implementations of such a complicated project.

6. Future Work

The current results of this project can be improved by digging into potential differences between the original implementation and our implementation, as well as exploring other datasets and model architectures.

Further work could include adding the Adult dataset, as well as replicating the other models from the original paper, DRO and IPW.

As noted in the original paper, the COMPAS dataset is known to be a noisy dataset, with label-bias or incorrect ground-truth labels. The original paper found that when the ARL model trained on the COMPAS dataset, the protected groups were not computationally-identifiable, and therefore produced a lower AUC score than the Baseline model. The ARL model struggles to perform with noisy outliers' such as those in the COMPAS dataset. Future work can improve upon identifying computationally-identifiable regions even with noisy data to improve the subpar AUC performance on datasets like COMPAS.

7. Conclusion

The original paper demonstrates how ARL can improve the fairness of ML algorithms when protected features are unknown or are not used. In our reimplementation, we were able to reproduce a similar trend in results, however we failed to reproduce the ARL improvement on the LSAC dataset. Throughout this process we learned how challenging data parsing can be, as well as how hard it can be to reproduce research results exactly, due to many unknown factors in the pipeline from data generation to metric evaluation. Future research in this area includes exploring other datasets and reevaluating model parameters and architectures.

As more research is done on ways to improve fairness in ML, especially in improving fairness without access to protected features, we hope that these novel methods will soon become embedded in the mainstream ML models that are used for countless tasks every day in the real world, deciding individuals' fates in the job market, at the bank, and in the courtroom.

8. Acknowledgement

We would like to acknowledge Professor Zoran Kostic, TAs : Mingyang Zang, Feroz Ahmad, Alexander Angus, Rachita Naik, and Adit Deshmukh, for their efforts throughout the semester. Everyone has been a great help to our group throughout the semester by answering our questions.

We would also like to acknowledge the authors, Alex Beutel, Jilin Chen, Kang Lee, Flavien Prost, Nithum Thain, and Xuezhi Wang, Ed H. Chi for their contributions to the original paper. The paper itself discussed an interesting challenge with a unique solution. Lastly, much thanks goes out to the online Keras documentation. One of the greatest challenges with this project was determining the proper functions and its specific details to reproduce the models. Keras documentation was very helpful in this process.

9. References

[1] S. Barocas and A. D. Selbst. 2016. Big data's disparate impact. Cal. L. Rev. 2016 (2016).
[2] S. Hajian, F. Bonchi, and C. Castillo. 2016. Algorithmic bias: From discrimination discovery to fairness-aware data mining. In SIGKDD.
[3] F. Kamiran, T. Calders, and M. Pechenizkiy. 2010. Discrimination Aware Decision Tree Learning. In ICDM
[4] M. Hardt, E. Price, and N. Srebro. 2016. Equality of Opportunity in Supervised Learning. In NIPS.
[5] M.B. Zafar, I. Valera, M. Rodriguez, K. Gummadi, and A. Weller. 2017. From parity to preference-based notions of fairness in classification. In NIPS
[6] J. Rawls. 2001. Justice as fairness: A restatement. Harvard University Press.
[7] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. 2012. Fairness through Awareness (ITCS '12). 214–226.
[8] R. S. Zemel, L. Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. 2013. Learning Fair Representations. In ICML
[9] P. Lahoti, K. P. Gummadi, and G. Weikum. 2019. Operationalizing Individual Fairness with Pairwise Fair Representations. VLDB 13, 4 (2019).
[10] P. Lahoti, Krishna P Gummadi, and G. Weikum. 2019. ifair: Learning individually fair data representations for algorithmic decision making. In ICDE. IEEE, 1334–1345.
[11] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian. 2015. Certifying and Removing Disparate Impact. In KDD.
[12] T. Kamishima, S Akaho, and J. Sakuma. 2011. Fairness-aware learning through regularization approach. In ICDMW.
[13] M. B. Zafar, I. Valera, M. Gomez-Rodriguez, and K. P. Gummadi. 2017. Fairness Beyond Disparate Treatment & Disparate Impact: Learning Classification without Disparate Mistreatment. In WWW.
[14] T. B. Hashimoto, M. Srivastava, H. Namkoong, and P. Liang. 2018. Fairness Without Demographics in Repeated Loss Minimization. In ICML

GitHub Link here.
(https://github.com/ecbme4040/e4040-2021fall-project-fair-krh2154-al4213-rg3332/tree/main)

10. Appendix

10.1 Individual Student Contributions in Fractions

	krh2154	al4213	rg3332
Last Name	Harwood	Lin	Gupta
Fraction of (useful) total contribution	1/3	1/3	1/3
What I did 1	Wrote all code: main notebooks, model architectures, data utils (except compas_input.py file), testing data stratification, including researching how to implement adversarial networks in keras	Developed flow charts of the overall code and models (primary, adversary, and baseline) architecture	Code for the data-utils/compas_input_data.py, experimented with different ways to embed categorical values and resolving their compatibility with the main notebooks
What I did 2	Produced train and test files from raw data, trained models, tuned hyperparams, produced AUC metrics	Conducted a comprehensive literature review of the research topic. Definition of fairness metrics, related research like DRO, etc.	Looked at the data distribution part & plots; did the literature survey/study and discussion for the project topic and parts like metric used.
What I did 3	Wrote Results section and created results tables, wrote most of Abstract, Conclusion, and contributed many paragraphs to other sections	Prepared the skeleton/start of most of the report. Wrote most of the introduction and implementation sections. Contributed paragraphs to other sections.	Contributed in writing sections of report, like conclusion, data, methodology of the original and this project (challenges, description etc); Reformatting of report

10.2 Support Material

Original Paper and Supplemental Material:
https://proceedings.neurips.cc/paper/2020/hash/07fc15c9d169ee48573edd749d25945d-Abstract.html

COMPAS Dataset:
https://www.propublica.org/datastore/dataset/compas-recidivism-risk-score-data-and-analysis

LSAC Dataset:
https://files.eric.ed.gov/fulltext/ED469370.pdf