Towards Fair Supervised Dataset Distillation for Text Classification

Xudong Han[1] **Aili Shen**[1,2*] **Yitong Li**[3] **Lea Frermann**[1]
Timothy Baldwin[1,4] **Trevor Cohn**[1]
[1]The University of Melbourne [2]Alexa AI, Amazon
[3]Huawei Technologies Co., Ltd. [4]MBZUAI
xudongh1@student.unimelb.edu.au aili.shen@amazon.com
liyitong3@huawei.com {lfrermann,tbaldwin,t.cohn}@unimelb.edu.au

Abstract

With the growing prevalence of large-scale language models, their energy footprint and potential to learn and amplify historical biases are two pressing challenges. Dataset distillation (DD) — a method for reducing the dataset size by learning a small number of synthetic samples which encode the information in the original dataset — is a method for reducing the cost of model training, however its impact on fairness has not been studied. We investigate how DD impacts on group bias in the context of text classification tasks, with experiments over two data sets, concluding that vanilla DD preserves the bias of the dataset. We then show how existing debiasing methods can be combined with DD to produce models that are fair and accurate, at reduced training cost.[1]

1 Introduction

Training and inference with deep neural networks is generally costly in terms of storage and computing resources. Model compression methods such as knowledge distillation (Hinton et al., 2015) and pruning (Cheong and Daniel, 2019) are popular ways of reducing model size to make inference more efficient. An alternative approach is dataset distillation ("DD": Wang et al. (2018)), which compresses the training set into a small synthetic dataset. Although there is significant pre-cost associated with DD in learning synthetic instances, DD has been shown to achieve almost identical performance to training over the original training set (Wang et al., 2018), at reduced computational cost. For example, Sucholutsky and Schonlau show that, on the IMDB binary sentiment classification dataset (Maas et al., 2011), a randomly initialized model trained over the distilled dataset with 10 instances per class achieves almost 97.8% of the original accuracy.

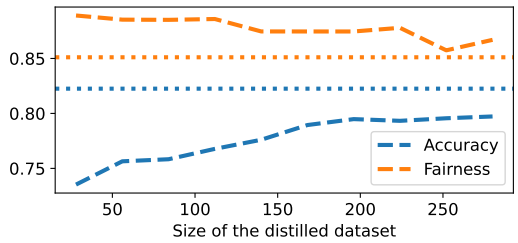

Figure 1: Performance and fairness over an occupation classification task, using models trained over the original (dotted lines) and distilled datasets (dashed lines) with different data sizes. For both metrics, larger is better. See Section 3.2 for full results.

It is well known that naively-trained models learn and amplify dataset biases, potentially leading to discrimination such as opportunity inequality (De-Arteaga et al., 2019). With the potential for compression methods to reduce training and storage cost, it is important to study their impact on model *fairness*. Here, we take DD as a case study and ask: (a) how does DD impact fairness; and (b) can we ensure fairness within this paradigm?

Specifically, we investigate how DD impacts on group fairness, and present experiments over two fairness benchmark datasets for text classification. Our contributions are: (1) we show that while DD preserves model performance, it also retains the dataset bias (e.g., Figure 1); and (2) we combine bias mitigation approaches with DD, and show that they improve fairness substantially.

2 Methodology

2.1 Dataset Distillation

Under STANDARD training, given inputs $\mathbf{x}$ annotated with main task labels $\mathbf{y}$ and protected labels $\mathbf{g}$, a neural network with parameters θ and loss function $\ell(\mathbf{x}, \mathbf{y}, \theta)$ learns $\theta^* = \arg\min_\theta \ell(\mathbf{x}, \mathbf{y}, \theta)$. Training with stochastic gradient descent (SGD) involves repeatedly sampling mini-batches of train-

*This work was done when Aili Shen was at The University of Melbourne.

[1]Source code available at https://github.com/HanXudong/Fair_Dataset_Distillation

Proceedings of The Third Workshop on Simple and Efficient Natural Language Processing (SustaiNLP), pages 65 - 72

ing data and updating network parameters based on their error gradient scaled by learning rate η, i.e., $\theta_{t+1} = \theta_t - \eta \nabla_{\theta_t} \ell\left(\mathbf{x}_t, \mathbf{y}_t, \theta_t\right)$.

With DD (Wang et al., 2018) the goal is to find a synthetic dataset such that SGD results in a parameter update that maximally improves the STANDARD training loss. This works by starting with initial parameters θ_0 and optimising:

$$\tilde{\mathbf{x}}^*, \tilde{\mathbf{y}}^*, \tilde{\eta}^* = \arg\min_{\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\eta}} \ell\left(\mathbf{x}, \mathbf{y}, \theta_0 - \tilde{\eta} \nabla_{\theta_0} \ell\left(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta_0\right)\right) \quad (1)$$

where the inner gradient term is the SGD update on synthetic data, and the outer-most loss is the STANDARD loss using the resulting parameter update. The synthetic data instances, $\tilde{\mathbf{x}}$, their labels, $\tilde{\mathbf{y}}$ (represented softly, using a softmax parameterisation; Sucholutsky and Schonlau (2021)), and the learning rate, $\tilde{\eta}$, are learned using gradient descent over Equation (1). This requires twice differentiable ℓ; see Wang et al. (2018) and Sucholutsky and Schonlau (2021) for full details of the training algorithm. The final step after learning this small synthetic dataset (typically in the realm of 10-100 examples) is to use it to train a new model, which we can then evaluate for accuracy (as in prior work), and fairness (unique to this work).

Note that the distillation computational cost scales positively with both the synthetic dataset size, and the size of the original training set. In Section 3.2, we provide the average runtime for DD. When retraining networks over the distilled instances, the original dataset will not be used and is irrelevant to the retraining cost.

2.2 Bias Mitigation

Previously proposed bias mitigation approaches can be classified as: (a) *pre-processing* the dataset before training (Wang et al., 2019; Han et al., 2021a); (b) adjusting the training algorithm itself *at-training* (Li et al., 2018; Shen et al., 2021); and (c) *post-processing* the trained models (Bolukbasi et al., 2016; Ravfogel et al., 2020), which is less relevant here, as it obscures the effect of DD itself. To learn fairer distilled datasets, we employ a pre-processing method and an at-training method *at distillation time*; no further debiasing is applied at model training. We compare performance and bias of a naively-trained model on distilled data (a) without debiasing and (b) with debiased distillation using one of the methods described next.

Balanced Training for Equal Opportunity Fairness (BTEO) Recall that DD learns the compressed synthetic training set from the original dataset. Intuitively, we would expect to learn a fairer synthetic dataset if the original dataset is balanced w.r.t. the classes and protected attribute(s), which we can achieve by pre-processing the dataset ($\mathbf{x}$, $\mathbf{y}$ and $\mathbf{g}$). BTEO implements equal opportunity fairness by balancing the distribution of protected attributes for each class (Han et al., 2021a).

We achieve the objective of BTEO by dataset downsampling, which essentially creates a balanced training set where each demographic group has the same number of training instances per class.

Adversarial Training (ADV) Following the setup of Elazar and Goldberg (2018); Li et al. (2018); Han et al. (2021c), the optimisation objective for standard adversarial training is:

$$\min_{\theta} \max_{\phi} \ell(\mathbf{y}, \hat{\mathbf{y}}) - \lambda \ell(\mathbf{g}, \hat{\mathbf{g}}) \quad (2)$$

where ϕ denotes the trainable parameters of the adversary, and λ is a trade-off hyperparameter. Solving this minimax optimization problem encourages the main task model hidden representations to be informative w.r.t. $\mathbf{y}$ but uninformative w.r.t. $\mathbf{g}$.

In terms of the ADV for DD debiasing, the optimization for DD (Equation (1)) is combined with the adversarial loss (Equation (2)), resulting in the minimax problem,

$$\min_{\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \tilde{\eta}} \max_{\phi} [\ell\left(\mathbf{x}, \mathbf{y}, \theta_0 - \tilde{\eta} \nabla_{\theta_0} \ell\left(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta_0\right)\right) - \lambda \ell(\mathbf{g}, \hat{\mathbf{g}})] \quad (3)$$

Similar to STANDARD +ADV, DD +ADV trains ϕ to predict $\hat{\mathbf{g}}$ over the final hidden representations of the real dataset ($\mathbf{x}$ and $\mathbf{g}$) extracted from the classifer. However, for DD, the classifier is trained over the synthetic datasets ($\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$). To decouple the training of model and discriminator, instead of solving the minimax problem with a gradient reversal layer (Ganin et al., 2016), we employ a two-step update following Han et al. (2021b). The negative sign for the adversarial loss ensures that adversary gradients are incorporated into DD to remove information related to protected attributes from the synthetic instances. For full details, see lines 6–9 of Algorithm 1 in Appendix A.

3 Experiments

In this section, we report experimental results for DD without and with debiasing. In Appendix B,

Model	Moji			Bios		
	Accuracy↑	**Fairness↑**	**DTO↓**	**Accuracy↑**	**Fairness↑**	**DTO↓**
Standard	72.3 ± 0.5	61.2 ± 1.4	47.7	82.3 ± 0.2	85.1 ± 0.8	23.2
Standard + BTEO	75.4 ± 0.1	87.7 ± 0.4	27.5	**83.8** ± 0.2	90.5 ± 0.9	**18.7**
Standard + ADV	75.6 ± 0.7	**89.3** ± 0.6	**26.6**	81.7 ± 0.2	**90.7** ± 0.8	20.5
DD	71.3 ± 1.8	62.4 ± 5.9	47.3	79.7 ± 0.4	86.7 ± 1.4	24.3
DD + BTEO	**75.7** ± 0.4	88.8 ± 1.1	26.8	73.2 ± 3.2	**90.7** ± 1.3	28.3
DD + ADV	72.8 ± 1.5	70.7 ± 9.1	40.0	80.3 ± 0.5	87.7 ± 1.2	23.2

Table 1: Evaluation results ± standard deviation (%) on the test set of sentiment analysis (Moji) and biography classification (Bios) tasks, averaged over 5 runs with different random seeds.

we provide full experimental details.

3.1 Experiment Setup

Dataset: We consider two tasks: (1) binary sentiment analysis over the Moji dataset (Blodgett et al., 2016), with protected "race" attributes (African American English vs. Standard American English); and (2) 28-way occupation classification with protected attribute gender (Male vs. Female) for each biography (De-Arteaga et al., 2019).

Text DD: In order to perform text DD, we follow Sucholutsky and Schonlau (2021) in learning synthetic samples from the embedding space. Specifically, we create the training set by extracting document embeddings from a fixed pretrained language model, such that the learned synthetic 'documents' are vectors rather than text inputs.

Models: Since the inputs to DD are document representations from a pretrained model, we follow the typical classification head setting (Felbo et al., 2017; Devlin et al., 2019) in using a multi-layer perceptron classifer as the model (θ) for DD that is trained over distilled instances $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$. Note that the parameter initialization is assumed to be known in this paper, which is the basic setting in Sucholutsky and Schonlau (2021). Specifically, θ_0 values are randomly sampled from Xavier Normal distribution (Glorot and Bengio, 2010), and are then repeatedly used in learning synthetic datasets and retraining the model over distilled datasets.

Evaluation Metrics: Following Ravfogel et al. (2020), we use overall accuracy as the performance metric, and the equal opportunity criterion (Hardt et al., 2016) to measure fairness in the form of the absolute recall differences (RD) between demographic groups. For ease of exposition, we report fairness as 1−RD, where larger is better and a perfectly fair model will achieve a score of 1.

In addition to reporting performance and fairness metrics separately, we also report distance to the optimal point ("DTO"), which quantifies the accuracy–fairness tradeoff (Marler and Arora, 2004; Han et al., 2021a). DTO measures the normalized Euclidean distance for a given combination of accuracy and fairness to the optimal point which denotes the ideal result, e.g., accuracy and fairness of 1.0. It is typically unachievable in practice.

Training Details: We follow Sucholutsky and Schonlau (2021) in our hyperparameter settings for text DD. Specifically, we conduct distillation for 10 GD steps, with 3 epochs for each iteration. Within each step, we generate one synthetic text embedding per target class, resulting in a total of 20 (= 10 steps × 2 classes) and 280 (= 10 steps × 28 classes) synthetic embeddings for Moji and Bios, respectively.

3.2 Results and Analysis

Table 1 summarizes the experimental results. Standard model is trained over the original training set without debiasing, while DD denotes models with the same architecture as Standard but trained over the distilled synthetic dataset. Over both datasets, DD achieves similar accuracy to Standard, consistent with previous work (Wang et al., 2018; Sucholutsky and Schonlau, 2021).

How does DD impact fairness? Similar patterns are observed over both datasets that fairness improves marginally: models trained over the distilled datasets retained 97.9% and 89.2% of the bias for Moji and Bios, respectively.

Can we ensure fairness of DD? As described in Section 2.2, we combined DD with two debiasing methods: BTEO (Han et al., 2021a) and ADV (Li et al., 2018). The methods are used only at the distillation stage, and not when training models over

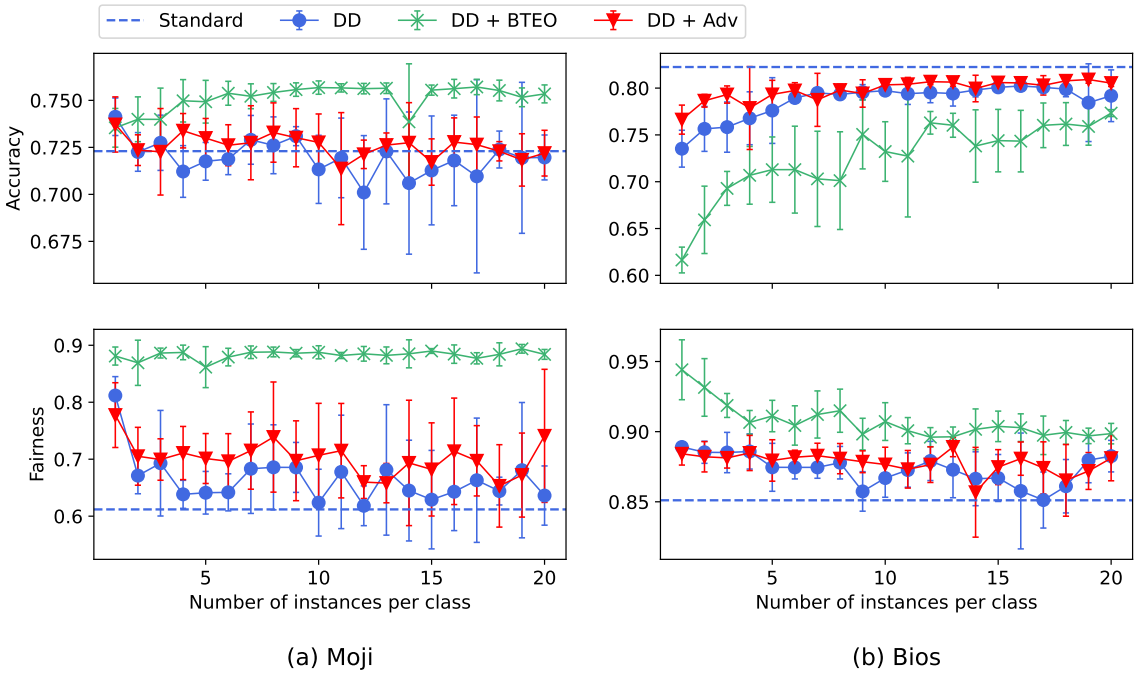

Figure 2: Evaluation results ± standard deviation with respect to different distilled dataset sizes.

the distilled dataset. Table 1 shows that, over the MOJI dataset, both STANDARD +BTEO and STANDARD +ADV improve fairness while also achieving better accuracy, leading to better performance–fairness trade-off (smaller DTO). This is consistent with previous work (Han et al., 2021c). Both debiasing methods substantially improve fairness while retaining accuracy compared to DD, with DD +BTEO being most effective.

In terms of BIOS, the fairness of DD +BTEO improves while accuracy drops appreciably, combining to result in a worse DTO. Since BIOS is a multi-class dataset with label skew, this is largely due to the naive sampling strategy of BTEO. Specifically, as suggested by Sucholutsky and Schonlau (2021), DD is less efficient for complex tasks, and we hypothesise the number of distilled instances for BIOS is insufficient for BTEO which additionally prevents the classifier from leaning unwanted correlations by manipulating the training dataset. DD +ADV, on the other hand, also improves fairness while retaining similar accuracy to DD.

Varying the size of distilled dataset To better understand the influence of the number of instances per class (= steps) in DD without explicit debiasing, we vary the number of steps from 1 to 20 (Figure 2). As the number of instances per class decreases, the accuracy drops substantially over BIOS, but stays relatively constant over MOJI, again implying that BIOS is a more challenging dataset than MOJI, due to the combination of the much larger label set and skew. Fairness scores generally increase as the number of instances per class decreases, but the combined DTO results are below the debiasing approaches at the same fairness level.

As for DD +BTEO over the BIOS set, the accuracy increases monotonically as the distilled dataset size increases, confirming our previous hypothesis that 10 instances per class is insufficient for BTEO.

Ultimately, the best choice of distilled dataset size is influenced by both the original task and the debiasing methods, and an important hyperparameter for DD debiasing that needs to be tuned jointly. For MOJI, an overly-large number of instances per class leads to worse performance and instability, while the harder task BIOS is more stable than MOJI given the same conditions, consisting with the previous work (Wang et al., 2018).

Training cost of DD Similar to pre-training, DD incurs a one-time cost in generating the distilled dataset. However, the computational cost of DD can be much more expensive than training the same model over the original datasets. Tables 2a and 2b show the computational cost for DD in seconds over MOJI and BIOS, respectively.

Table 3 compares the training time for the non-DD methods with DD (with 10 instances per class) over the two datasets. ADV incurs additional cost

Size	Distillation			Train
	DD	DD + ADV	DD + BTEO	
1	144	139	74	< 0.5
2	195	203	95	< 0.5
3	252	259	115	< 0.5
4	307	302	141	< 0.5
5	377	352	165	< 0.5
6	442	411	181	1
7	477	463	212	1
8	531	524	227	1
9	581	621	248	1
10	632	629	268	1
11	716	722	292	1
12	770	774	314	1
13	840	864	345	1
14	882	852	363	1
15	931	923	408	2
16	961	1051	417	2
17	998	991	438	2
18	1064	1052	473	2
19	1124	1158	488	2
20	1162	1268	525	2

(a) MOJI

Size	Distillation			Train
	DD	DD + ADV	DD + BTEO	
1	474	462	108	< 0.5
2	783	756	153	1
3	1137	1107	199	1
4	1459	1578	249	1
5	1919	1856	292	1
6	2325	2208	346	1
7	2485	2462	394	1
8	2802	2876	447	2
9	3037	3114	502	2
10	3453	3848	508	2
11	3769	3862	577	3
12	4225	4179	647	3
13	4620	4607	694	4
14	5370	4799	741	4
15	5084	5097	865	4
16	5442	6012	824	4
17	6139	6399	865	5
18	6007	6171	938	5
19	6685	6566	937	5
20	6728	6971	1025	5

(b) BIOS

Table 2: Computational cost (sec) to: (a) learn synthetic instances through distillation; and (b) train the model over the synthetic instances.

	Training Time	
Model	MOJI	BIOS
STANDARD	35	96
STANDARD + ADV	40	135
STANDARD + BTEO	19	32
DD	1	2

Table 3: Training time (sec) over MOJI and BIOS.

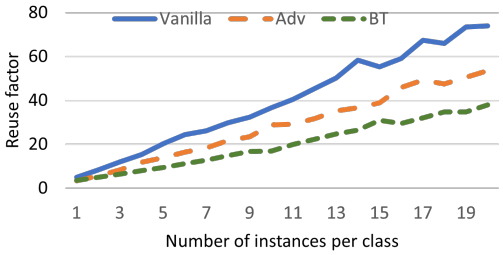

Figure 3: Reuse factor of DD on the BIOS dataset.

over STANDARD due to the discriminator training, while BTEO results in faster training due to the reduction in training set size.

In terms of DD, the debiasing is only employed as part of learning the synthetic instances, and does not affect the classifier training over the distilled dataset. As such, the training time for DD, DD + ADV, and DD + BTEO is identical. As it can be seen, training a model over a pre-distilled dataset is much faster than the STANDARD training, but when the combined cost of dataset distillation and model training is taken into account, it is around an order of magnitude slower than STANDARD training.

To further analysis how the upfront task of DD compares to the training cost, Figure 3 shows the reuse factor w.r.t. different DD sizes. For example, for a naively trained model, the reuse factor for size 10 is calculated as $\frac{3453}{96-2} \approx 36.73$, where 3453 and 2 are the distillation time and training time of DD w.r.t. size 10 (Table 2b), and 96 is the training time of STANDARD (Table 3). It can be seen that the ratio of neutralization and distilled dataset size are positively correlated, and debiasing methods (ADV and BTEO) requires less time to neutralize the pre-cost of DD.

4 Conclusion

This paper evaluated the effect of dataset distillation on fairness in the context of two text classification tasks. Empirically, we showed that distilled datasets retain unwanted biases. In order to learn fairer synthetic datasets, we employ adversarial learning and balanced training for bias mitigation, which results in substantial fairness improvements. We conclude that DD can be effective for fair *and* efficient text classification, specifically for simpler tasks where a small distilled data set is sufficient.

Acknowledgements

We thank the anonymous reviewers for their helpful feedback and suggestions. This work was funded by the Australian Research Council, Discovery grant DP200102519. This research was undertaken using the LIEF HPC-GPGPU Facility hosted at the University of Melbourne. This Facility was established with the assistance of LIEF Grant LE170100200.

Limitations

Hyperparameter Tuning Taking ADV debiasing as an example, the current practice is to fine-tune the trade-off hyperparameter to find the best-performing model. However, tuning the trade-off hyperparameter is too expensive for DD due to the high cost of distillation. In this paper, we assume that the trade-off hyperparameters for bias mitigation are only affected by the training dataset and model architecture. Based on this assumption, we adopted the best trade-off hyperparameter settings from STANDARD training for DD, but further exploration of this interaction is warranted in future work.

Parameter Initialization We adopted the DD framework of Sucholutsky and Schonlau (2021), including its strong assumptions about the initial parameters of the classifiers that are trained over the synthetic datasets. Specifically, the initial weights (θ_0) are assumed to be either fixed and known, or drawn from a fixed and known distribution, before and after distillation. This implies, for example, that the classifier architecture has to be the same as what was used during distillation. However, this assumption underlies most existing DD work, and is outside the scope of this research.

Ethical Considerations

This work aims to detect and mitigate bias in distilled datasets in NLP. For DD, both the distillation over original datasets and the classifier training over the distilled datasets do not access the protected attributes. However, consistent with previous work, the fairness definition and evaluation are based on the protected attributes. Briefly, the protected attributes are unobserved during distillation, model training, and inference, and are only used for evaluation purposes.

This work relies on benchmarks to evaluate model fairness and accuracy. Like much previous work, these benchmarks propose an oversimplified, binary notion of the protected attributes. We acknowledge that both gender (Sun et al., 2019) and race (Field et al., 2021) are more nuanced. As a result of such oversimplification specifically, and the controlled nature of benchmarks in general, we recommend to additionally consult user studies and application scenarios to obtain holistic measure of model fairness.

In terms of the DD debiasing, the employed method requires access to training datasets with protected attributes, consistent with previous work on adversarial training and other bias mitigation methods. After distillation, it is important to note that the model trained over the debiased distilled dataset can make fairer predictions without any requirement of demographic information for either synthetic instances or test instances.

We only use attributes that the user has self-identified in our experiments. All data in this study is publicly available and used under strict ethical guidelines.

References

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. 2016. Demographic dialectal variation in social media: A case study of African-American English. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1119–1130.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*, pages 4349–4357.

Robin Cheong and Robel Daniel. 2019. transformers.zip: Compressing transformers with pruning and quantization. *Technical report, Stanford University, Stanford, California*.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. 2019. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 120–128.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186.

Yanai Elazar and Yoav Goldberg. 2018. Adversarial removal of demographic attributes from text data. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 11–21.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Anjalie Field, Su Lin Blodgett, Zeerak Waseem, and Yulia Tsvetkov. 2021. A survey of race, racism, and anti-racism in NLP. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1905–1925, Online. Association for Computational Linguistics.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021a. Balancing out bias: Achieving fairness through training reweighting. *arXiv preprint arXiv:2109.08253*.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021b. Decoupling adversarial training for fair NLP. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 471–477.

Xudong Han, Timothy Baldwin, and Trevor Cohn. 2021c. Diverse adversaries for mitigating bias in training. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2760–2765.

Xudong Han, Aili Shen, Yitong Li, Lea Frermann, Timothy Baldwin, and Trevor Cohn. 2022. fairlib: A unified framework for assessing and improving classification fairness. *arXiv preprint arXiv:2205.01876*.

Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in Neural Information Processing Systems*, 29:3315–3323.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards robust and privacy-preserving text representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

R Timothy Marler and Jasbir S Arora. 2004. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7237–7256.

Aili Shen, Xudong Han, Trevor Cohn, Timothy Baldwin, and Lea Frermann. 2021. Contrastive learning for fair representations. *arXiv preprint arXiv:2109.10645*.

Ilia Sucholutsky and Matthias Schonlau. 2021. Soft-label dataset distillation and text dataset distillation. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1630–1640, Florence, Italy. Association for Computational Linguistics.

Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. 2019. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5310–5319.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. 2018. Dataset distillation. *arXiv preprint arXiv:1811.10959*.

Algorithm 1 Fair Dataset Distillation

Input: θ_0 = initial weights of the main model; α = DD step size; T = number of optimization iterations; $\tilde{y}_0$ = initial value for $\tilde{y}$; $\tilde{\eta}_0$ = initial value for $\tilde{\eta}$; λ = strength of the adversarial regularization

Initialize distilled dataset with M instances $\tilde{\mathbf{x}} = \{\tilde{x}_i\}_{i=1}^{M}$ randomly,
$\tilde{\mathbf{y}} = \{\tilde{y}_i\}_{i=1}^{M} \leftarrow \tilde{y}_0$,
$\tilde{\eta} \leftarrow \tilde{\eta}_0$
for each training step t = 1 to T **do**
  Get a mini-batch of n real training instances $(\mathbf{x}_t, \mathbf{y}_t) = \{x_{t,j}, y_{t,j}\}_{j=1}^{n}$
  Compute updated model parameters with GD $\theta_1 = \theta_0 - \tilde{\eta}\nabla_{\theta_0}\ell\left(\tilde{\mathbf{x}}, \tilde{\mathbf{y}}, \theta_0\right)$
  Evaluate the objective function on real training data: $\mathcal{L} = \ell\left(\mathbf{x}_t, \mathbf{y}_t, \theta_1\right)$
  if ADV **then**
    Update the discriminator ϕ $\phi = \phi - \eta_{\text{adv}}\nabla_{\phi}\ell\left(\mathbf{x}_t, \mathbf{g}_t, \theta_1, \phi\right)$
    Incorporate adversarial loss $\mathcal{L} = \mathcal{L} - \lambda\ell\left(\mathbf{x}_t, \mathbf{g}_t, \theta_1, \phi\right)$
  end if
  Update distilled data $\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}} - \alpha\nabla_{\tilde{\mathbf{x}}}\sum_j \mathcal{L}$, $\tilde{\mathbf{y}} \leftarrow \tilde{\mathbf{y}} - \alpha\nabla_{\tilde{\mathbf{y}}}\sum_j \mathcal{L}$, $\tilde{\eta} \leftarrow \tilde{\eta} - \alpha\nabla_{\tilde{\eta}}\sum_j \mathcal{L}$
end for

Output: distilled data $\tilde{\mathbf{x}}$; labels $\tilde{\mathbf{y}}$; and learning rate $\tilde{\eta}$

A Algorithm

The algorithm for combined DD with debiasing techniques is detailed in Algorithm 1.

B Experimental Details

B.1 Dataset Splits

We use the same data split as previous work (Ravfogel et al., 2020), resulting in train, dev, and test splits of 100k/8k/8k for MOJI, and 257k/40k/100k for BIOS.

B.2 Fairness Metric

We follow the previous work (Han et al., 2021c) in reporting the RMS recall (TPR) disparities. The calculation of RMS TPR GAP consists of aggregations at the group and class levels. At the group level, we measure the absolute TPR difference of each class between each group and the overall TPR $GAP_{G,y}^{TPR} = \sum_{g \in G} |TPR_{g,y} - TPR_y|$, and at the class level, we further perform the RMS aggregation at the class level to get the RMS TPR GAP as $GAP = \sqrt{\frac{1}{|Y|}\sum_{y \in Y}(GAP_{G,y}^{TPR})^2}$.

B.3 Models

We follow Ravfogel et al. (2020) in using DeepMoji (Felbo et al., 2017) as the encoder to get 2304d representations of the input texts. For the BIOS dataset, we follow Han et al. (2021a) in taking 768d 'AVG' representations from BERT-base (Devlin et al., 2019), which takes the average of all contextualized token embeddings.

The number of trainable parameters of the classifier is about 1M for both tasks. The synthetic datasets can also be treated as trained parameters, and the total number of trainable parameters are influenced by, dimension of the embedding space (nd), the number of classes (nc), and the number of distillation steps (ns), resulting in $ns \times (nd \times nc + nc \times nc + 1$, which are $4613 \times ns$ and $22289 \times ns$ for MOJI and BIOS, respectively.

We notice that DD is slow to converge for some random initializations. When running DD with different random seeds, we set dataset-specific accuracy thresholds to filter unconverged runs, which are 0.65 and 0.6 for MOJI and BIOS, respectively.

B.4 Computing Infrastructure

We conduct our experiments on a Windows server with a 16-core CPU (AMD Ryzen Threadripper PRO 3955WX), two NVIDIA GeForce RTX 3090s with NVLink, and 256GB RAM, and on a HPC cluster instance with 4 CPU cores, 32GB RAM, and one NVIDIA V100 GPU.

B.5 Hyperparameter Tuning

We use the same model architectures and hyperparameters as previous work (Han et al., 2022), which are shown to achieve better results than the results in the original paper of BTEO (Han et al., 2021a) and ADV (Li et al., 2018). We vary the size of the distilled dataset from 1 to 20 for each method and run experiments 7 times with different random seeds for each hyperparameter combination. Corresponding scripts are included in the submitted code file.