FairBatch: Batch Selection for Model Fairness

Yuji Roh[1], Kangwook Lee[2], Steven E. Whang[1], and Changho Suh[1]
[1]KAIST, [2]University of Wisconsin-Madison

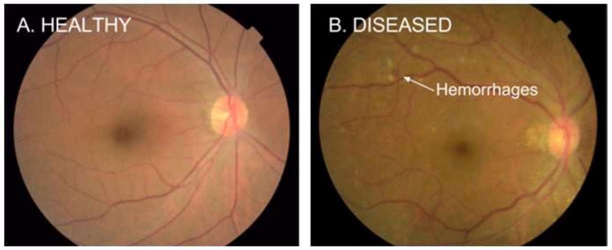
ICLR 2021

slides based on 'J. Lee, Y. Roh, H. Song, and S. E. Whang, Machine Learning Robustness, Fairness, and their Convergence (Tutorial), ACM SIGKDD 2021'

Deep Learning (DL)

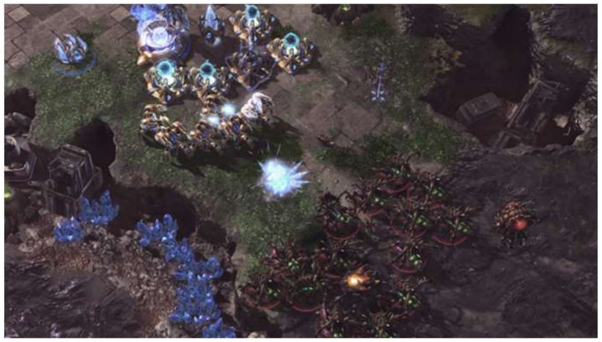
- DL is widely used to glean knowledge from massive amounts of data
- Applications: natural language understanding, healthcare, self driving cars, ...

<Google Translate>



<Diabetic Retinopathy>

<Self-driving car>



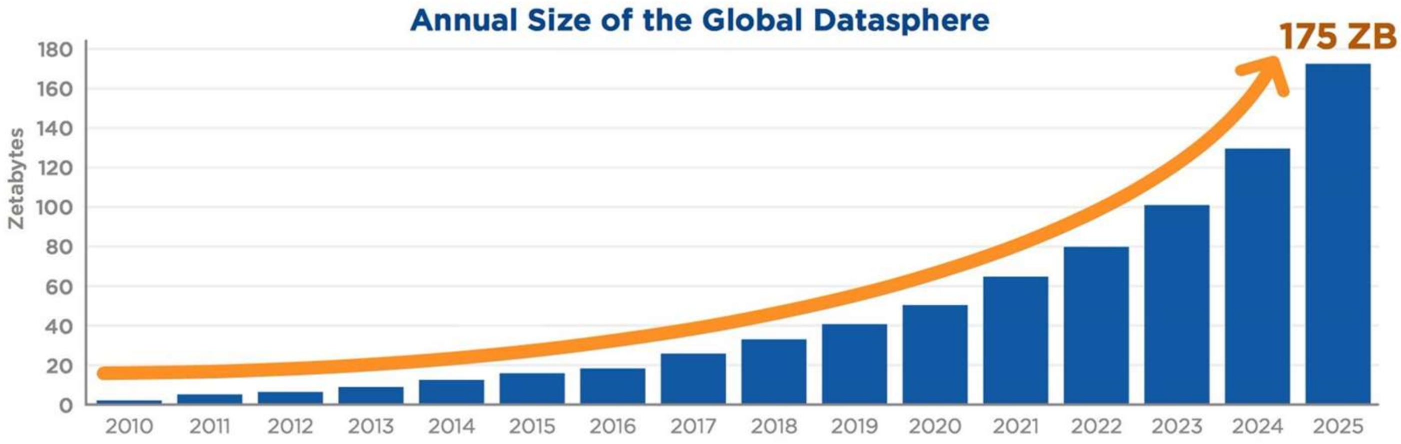
<AlphaStar>

Source: Google

Game Changer 1: Big Data

1 ZB = 1,000,000,000,000 GB
≈ 1 Great Wall of China

Source: Pixabay

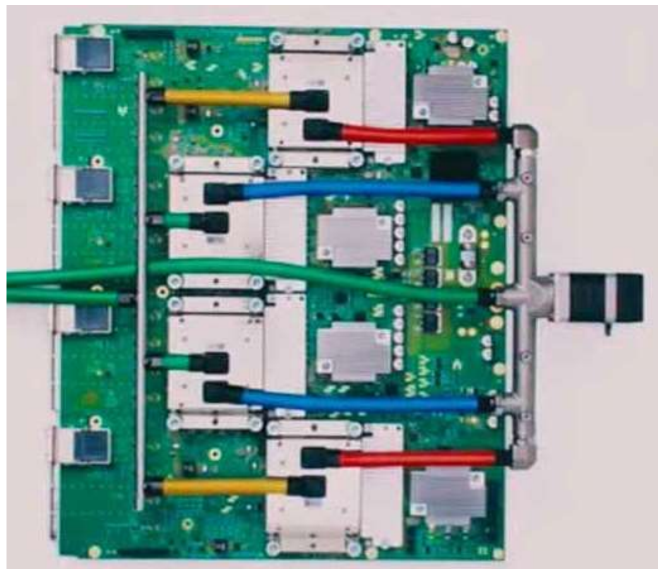


Source: IDC

Game Changer 2: Fast Computation

Tensor Processing Unit (TPU) 4.0

Source:Joel Garcia Jr

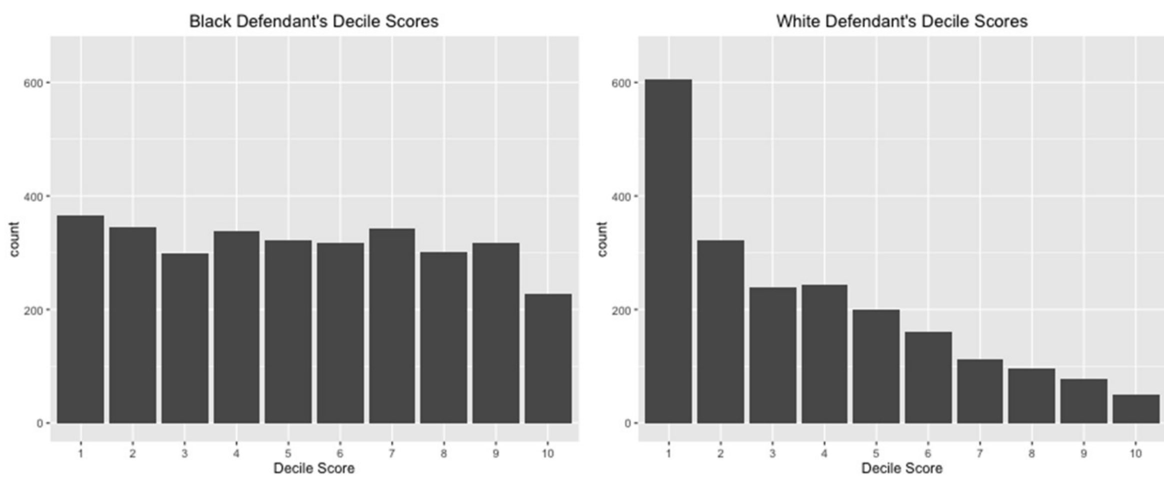


Source: Wikipedia

Source: The Verge

Fairness

- Along with the proliferation of AI applications, there has been a growing concern on gender, race, and other types of bias in these systems
    - Example: Severe bias against African Americans in COMPAS to score criminal defendants for recidivism risk

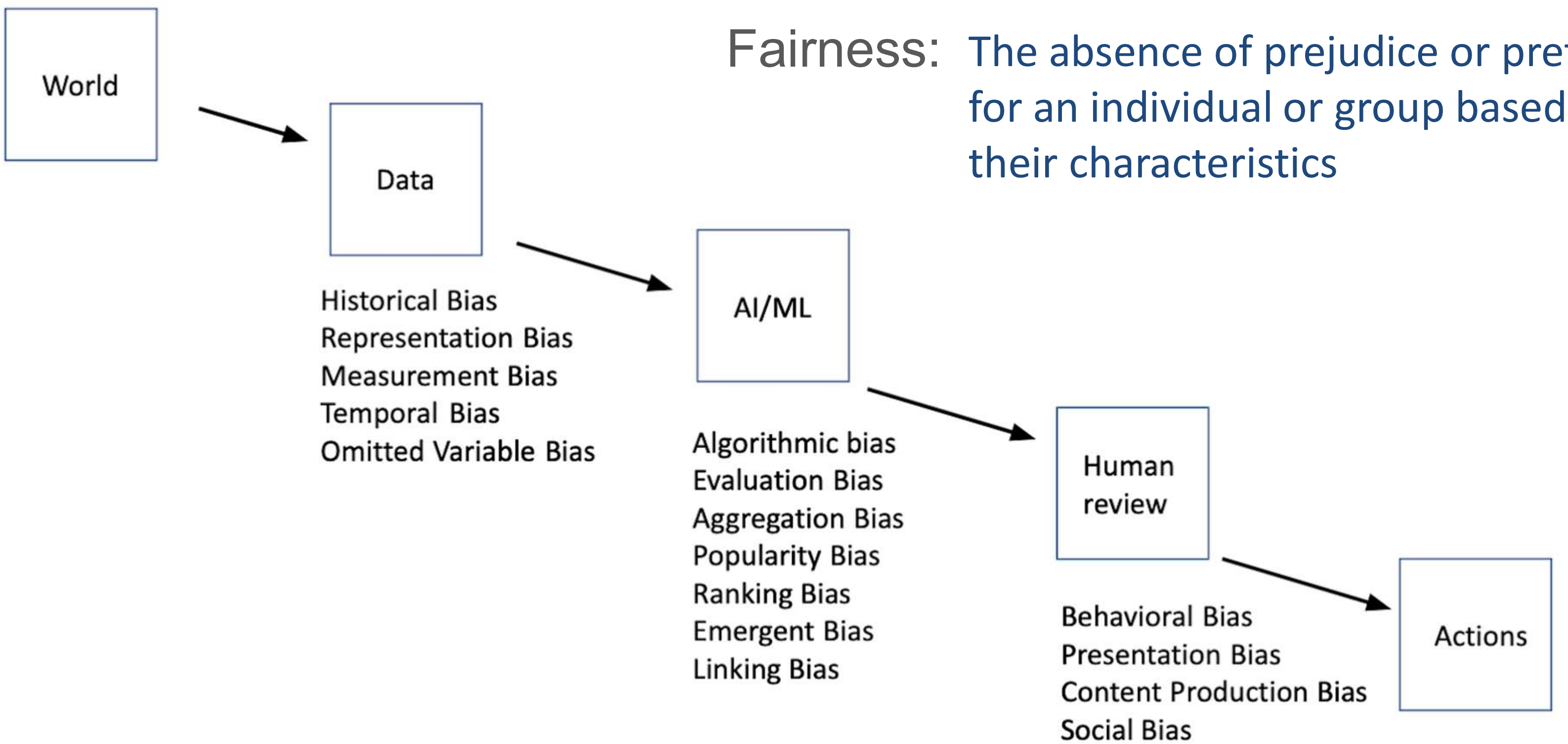


Source: https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm

Bias and Fairness

Fairness: The absence of prejudice or preference for an individual or group based on their characteristics

Bias:



Source: https://youtu.be/fA5xpRnqbKM

Fairness Criteria [Barocas et al., FairMLBook19]

- Most fairness measures fall into the following three criteria
  - *Independence*: $\hat{Y} \perp Z$ Demographic parity [Feldman et al., KDD15]
  - *Separation*: $\hat{Y} \perp Z | Y$ Equalized odds [Hardt et al., NeurIPS16]
  - *Sufficiency*: $Y \perp Z | \hat{Y}$ Predictive parity [Chouldechova, BigData17]

$\hat{Y}$ Model prediction Y True label Z Sensitive attribute

Demographic Parity

[Feldman et al., KDD15]

- Sensitive groups have same positive prediction rates
- Applications: predicting crime, hiring, and giving loans

$$P(\hat{Y} = 1 | Z = 0) = P(\hat{Y} = 1 | Z = 1)$$

Model predicts recidivism

White person

Black person

Equalized Odds

[Hardt et al., NeurIPS16]

- Sensitive groups have same positive prediction rates **when label = 0 or 1**
- Intuitively, distinguishes “qualified” from “unqualified” people

$$P(\hat{Y} = 1 | Z = 0, Y = A) = P(\hat{Y} = 1 | Z = 1, Y = A)\,, A \in \{0, 1\}$$

Model predicts recidivism

White person

Black person

Person actually committed crime

Predictive Parity [Chouldechova, BigData17]

- Sensitive groups have same positive **label** rates when **prediction** = 0 or 1
- Intuitively, the model's precision rates are similar for sensitive groups

$$P(Y = 1|Z = 0, \hat{Y} = A) = P(Y = 1|Z = 1, \hat{Y} = A),\ A \in \{0, 1\}$$

Person actually committed crime

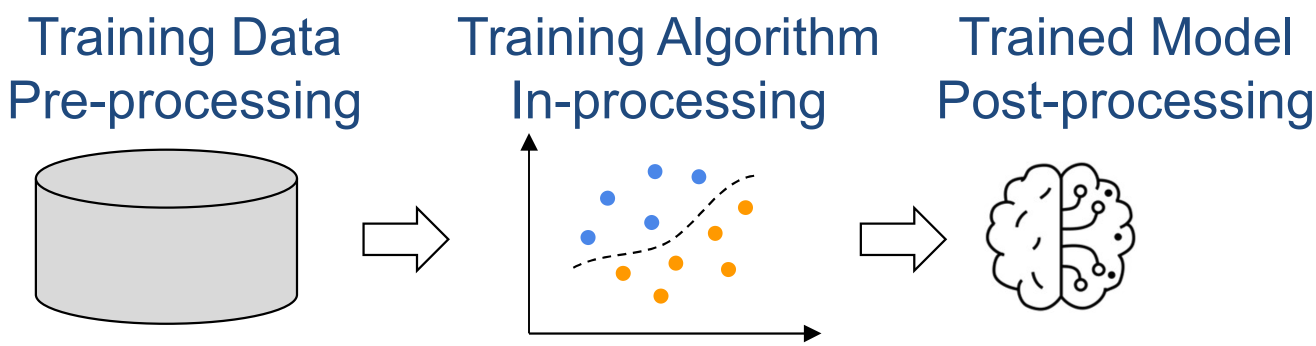
White person

Black person

Model predicts recidivism

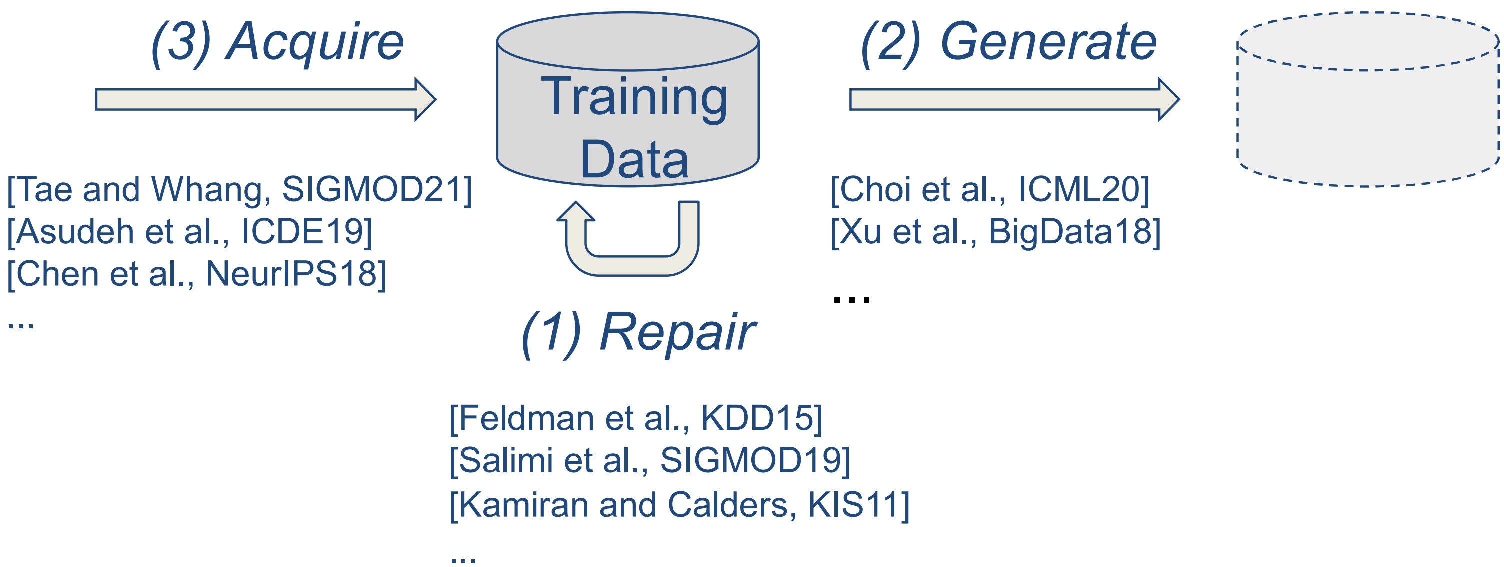
Unfairness Mitigation [Bellamy et al., CoRR18]

- Addressing data bias can be categorized into pre-/in-/post-processing
- Depends on whether bias is mitigated before/during/after model training



Pre-processing: Preparing Unbiased Data

- Approach: fix unfairness before model training
- Pros: can solve root cause of unfairness
- Cons: tricky to ensure model fairness actually improves



What Doesn't Work: Removing Sensitive Attributes

- Sensitive attributes are usually correlated with other attributes

age

name

income

credit score

race

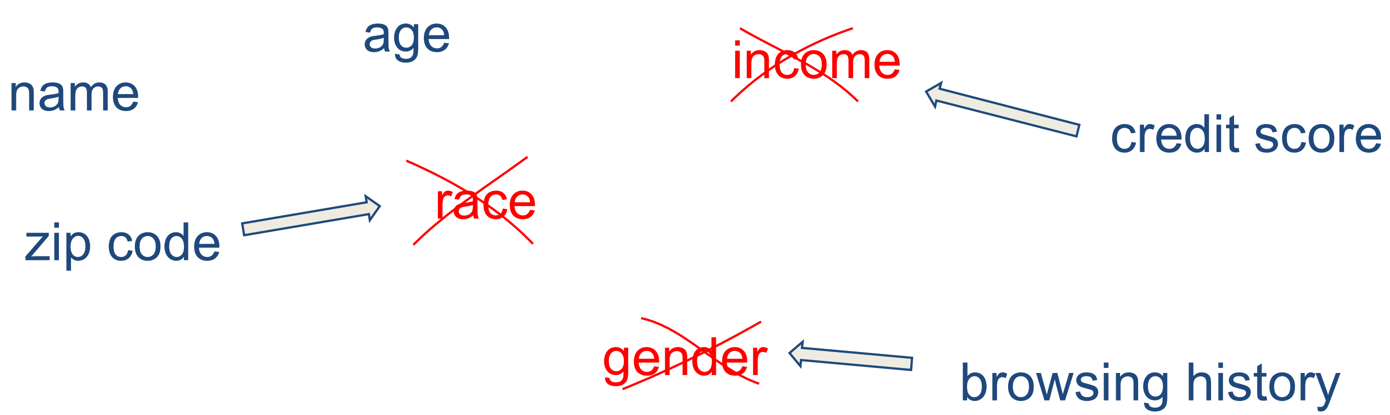
zip code

gender

browsing history

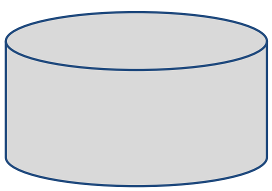
What Doesn’t Work: Removing Sensitive Attributes

- Sensitive attributes are usually correlated with other attributes



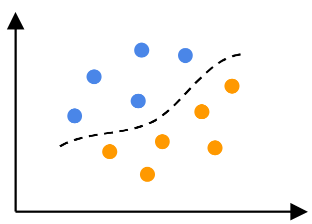
In-processing: Training on Biased Data

- Approach: fix model training for fairness
- Pros: can directly optimize accuracy and fairness
- Cons: may have to make significant changes in model training



(3) Adaptive reweighting

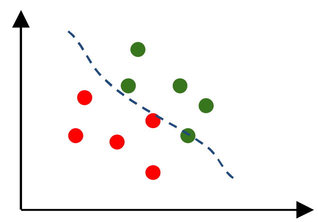
[Roh et al., ICLR21]
[Zhang et al., SIGMOD21]
[Iosifidis and Ntoutsi, CIKM19]
...



(1) Add fairness constraints

[Zafar et al., AISTATS17]
[Agarwal et al., ICML18]
[Kamishima et al., PKDD12]
...

(2) Adversarial training



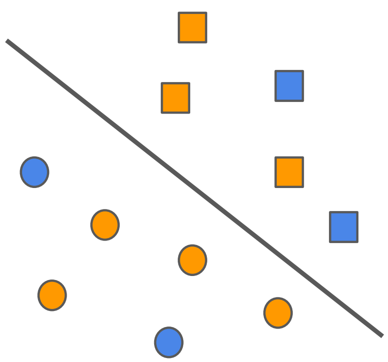
[Zhang et al., AIES18]
[Cotter et al., ALT19]
[Roh et al., ICML20]
...

Fairness Constraints [Zafar et al, AISTATS17]

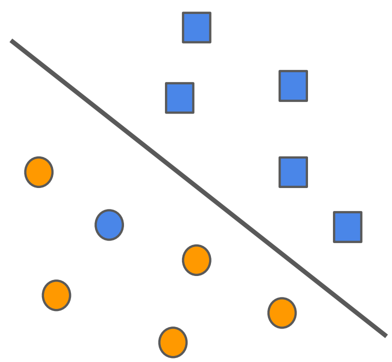
Maximize accuracy with fairness constraints for convex margin classifiers

- Want to satisfy demographic parity as much as possible: $P(\hat{Y}=1|Z=0) \approx P(\hat{Y}=1|Z=1)$
- However, this constraint is not convex, so use a proxy instead
- Proxy: covariance between sensitive attribute and signed distance to decision boundary

Good



Bad

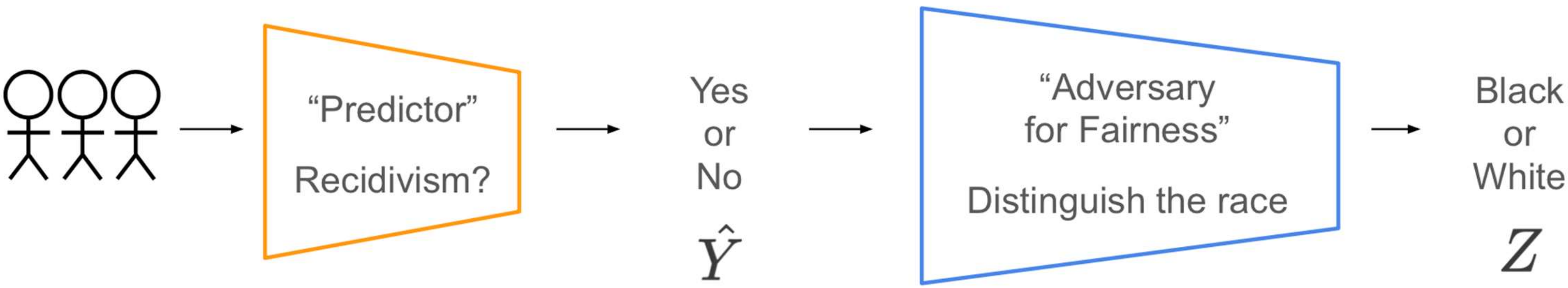


Intuition: sensitive attribute should not imply which side you are on the decision boundary (i.e., the label)

Adversarial Debiasing

[Zhang et al., AIES18]

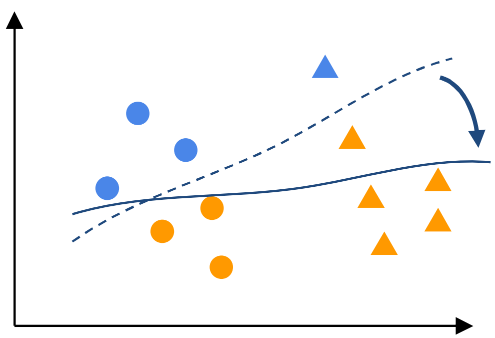
Compete: predictor (predict Y) and adversary (predict Z from $\hat{Y}$)



Demographic parity theorem (similar for other fairness criteria):
If adversary optimally predicts Z from $\hat{Y}$
and predictor completely fools adversary,
then $\hat{Y} \perp Z$, so $P(\hat{Y}=1|Z=0) = P(\hat{Y}=1|Z=1)$

Post-processing: Debiasing a Trained Model

- Approach: fix model predictions for fairness
- Pros: only option if data and model cannot be modified
- Cons: usually results in worse accuracy



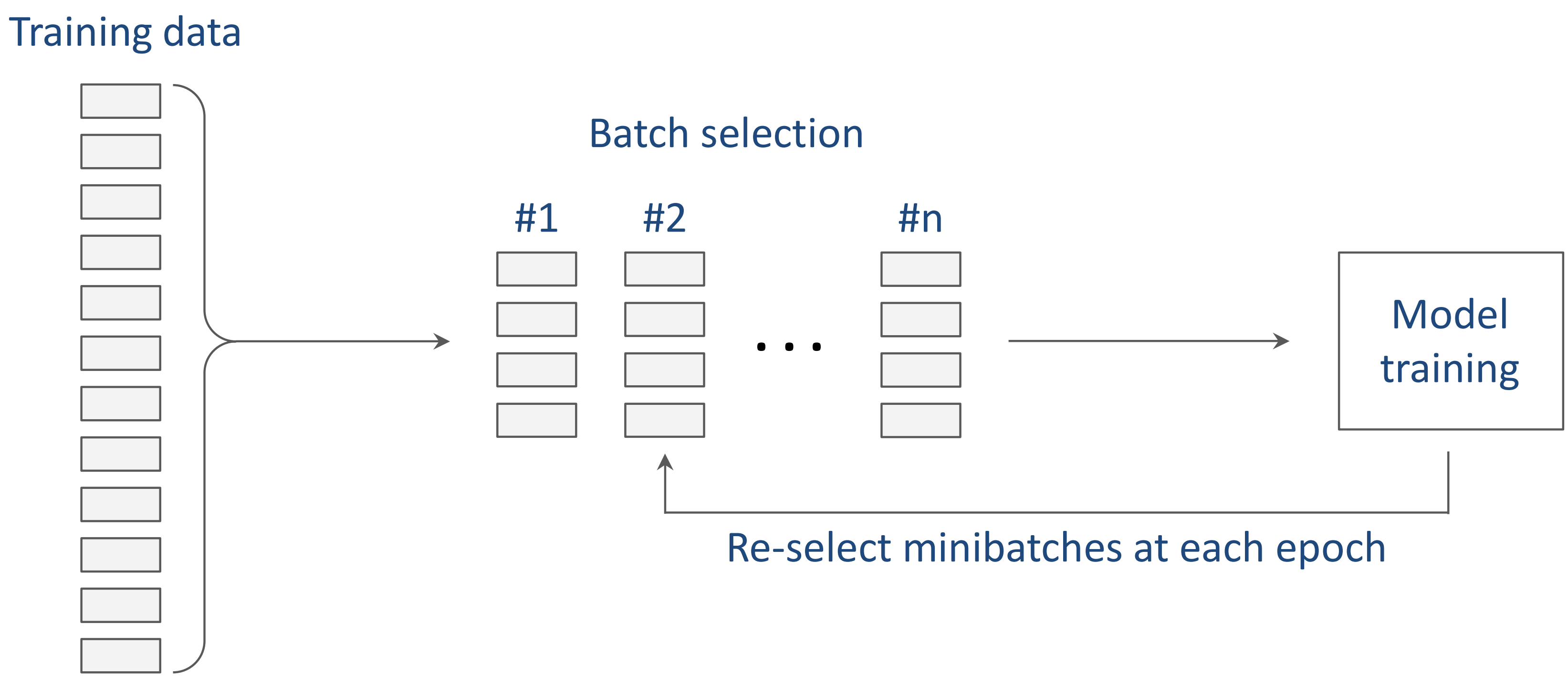
[Hardt et al., NeurIPS16]
[Chzhen et al., NeurIPS19]
[Pleiss et al., NeurIPS17]
...

This paper: FairBatch
Batch Selection for Model Fairness

- Idea: perform “fair” sampling during batch selection
- Categorized as in-processing, but actually does not modify training algorithm

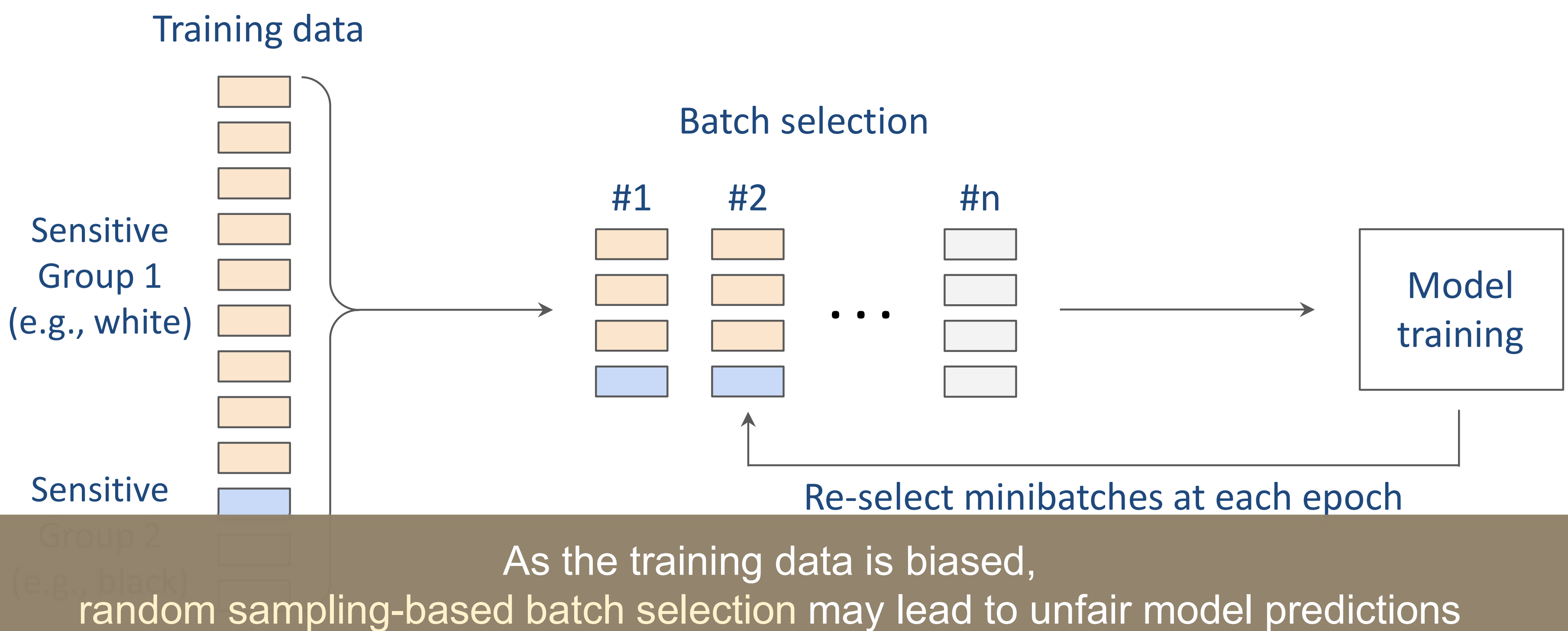
Batch Selection

- Model training commonly employs batch selection to get minibatches from training data
- Random sampling is often used to select minibatches



Batch Selection

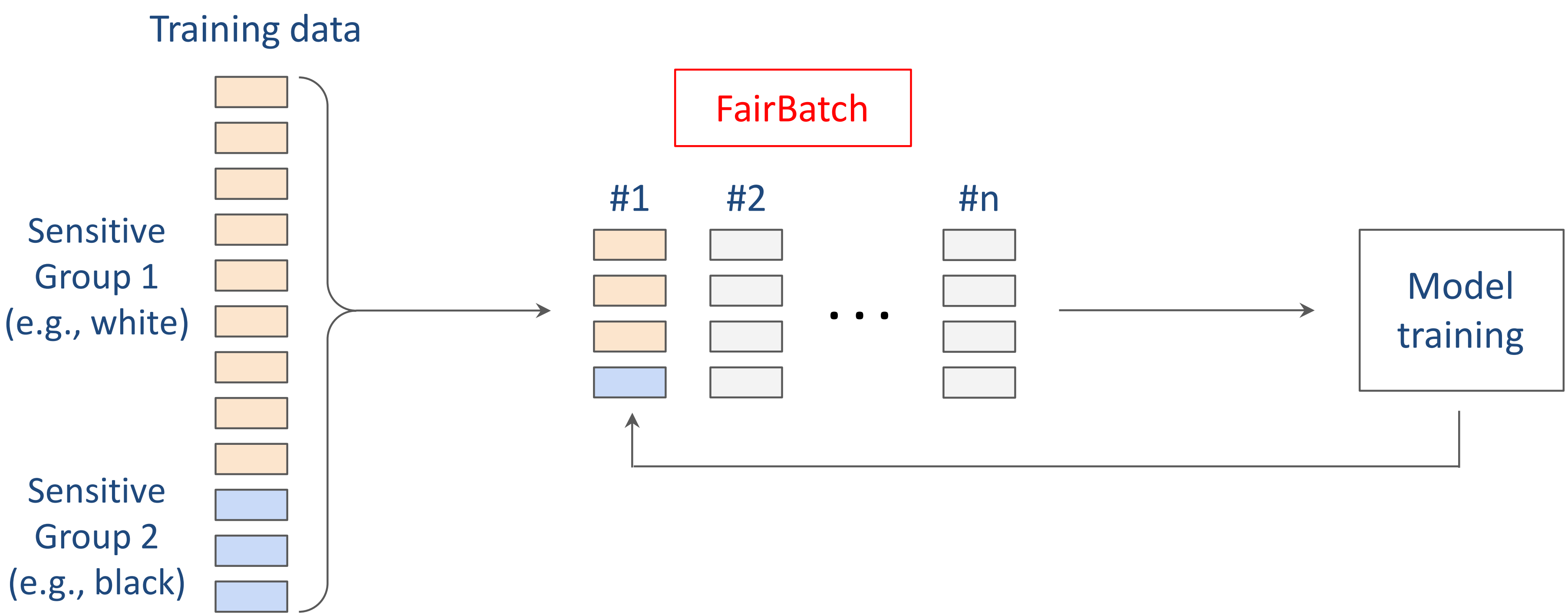
- Model training commonly employs batch selection to get minibatches from training data
- Random sampling is often used to select minibatches



As the training data is biased,
random sampling-based batch selection may lead to unfair model predictions

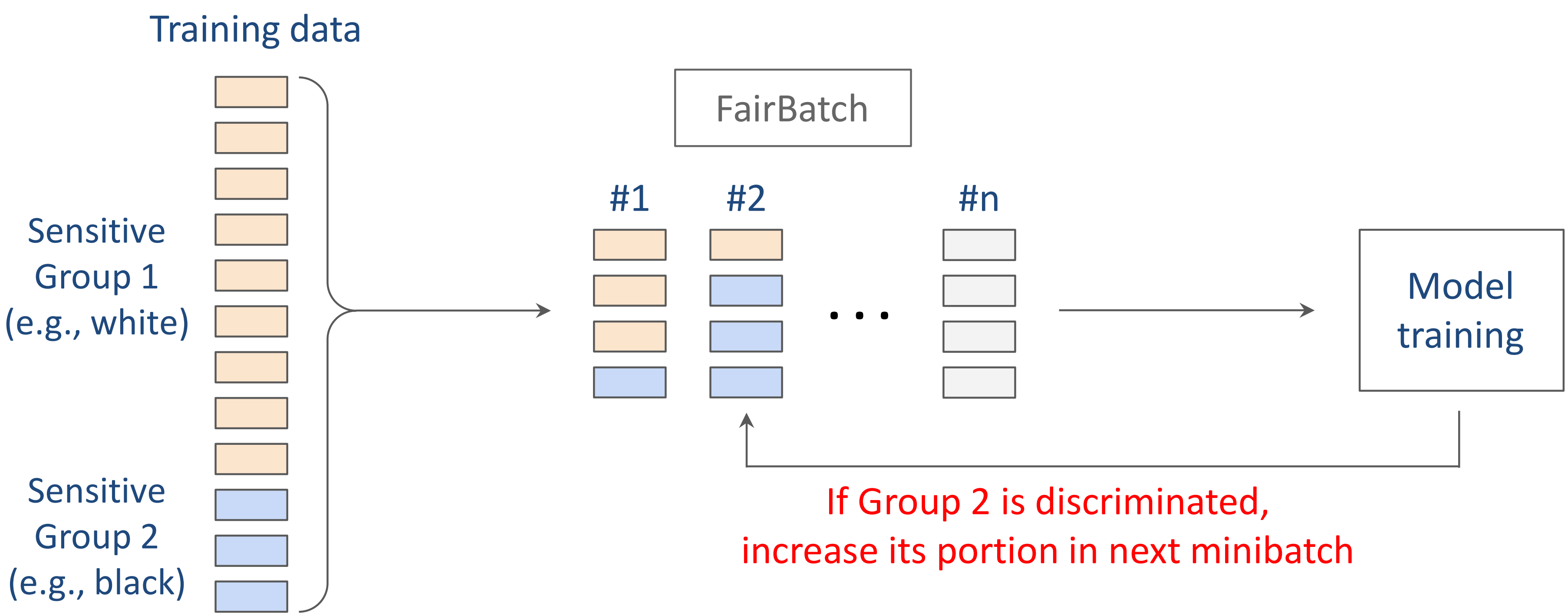
FairBatch: Batch Selection for Model Fairness

- Adaptively selects minibatch sizes for the purpose of improving model fairness
- Adjusts the sizes w.r.t. sensitive groups based on the fairness of intermediate models



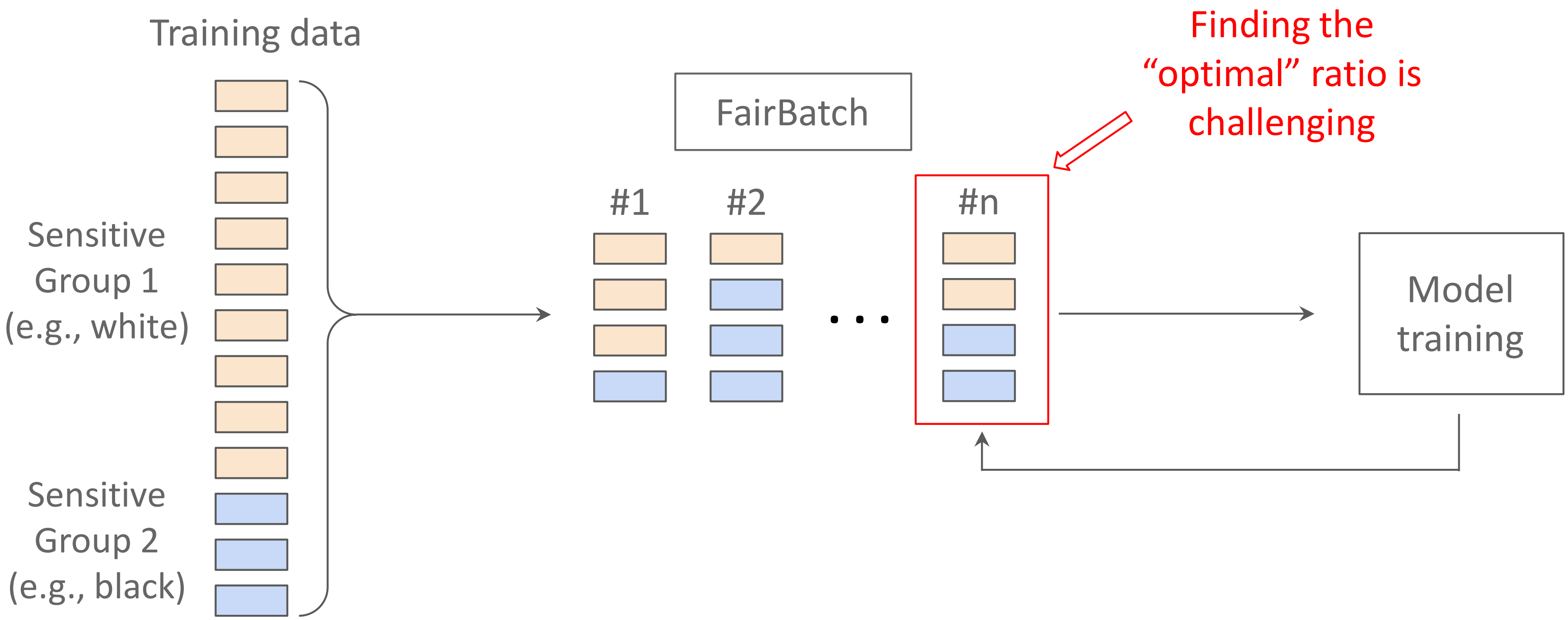
FairBatch: Batch Selection for Model Fairness

- Adaptively selects minibatch sizes for the purpose of improving model fairness
- Adjusts the sizes w.r.t. sensitive groups based on the fairness of intermediate models



FairBatch: Batch Selection for Model Fairness

- Adaptively selects minibatch sizes for the purpose of improving model fairness
- Adjusts the sizes w.r.t. sensitive groups based on the fairness of intermediate models



FairBatch: Batch Selection for Model Fairness

- Key features
  - **Adaptively selects minibatch sizes** for the purpose of improving model fairness
    - Solves bilevel optimization for fairness and accuracy
  - Can be employed with a **single-line** change of PyTorch code
- Performance results
  - Obtains high accuracy and fairness within one training
  - Runs **15 ~ 96x faster** than fair training baselines
  - Gracefully merges with existing batch selection techniques used for faster convergence

Problem formulation: ERM

$\boldsymbol{w}$: model parameter; $\mathrm{x} \in \mathbb{X}$: input feature; $\widehat{\mathrm{y}} \in \mathbb{Y}$: predicted class; $\mathrm{z} \in \mathbb{Z}$: sensitive attribute (e.g., gender)

Consider the 0/1 loss: $\ell(\mathrm{y}, \widehat{\mathrm{y}}) = 1(\mathrm{y} \neq \widehat{\mathrm{y}})$, and let m be the total number of train samples.

$L_{y,z}(\boldsymbol{w})$: the empirical risk aggregated over samples subject to $\mathrm{y} = y$ and $\mathrm{z} = z$;
The overall empirical risk is written as $L(\boldsymbol{w}) = \frac{1}{m}\sum_i \ell(\mathrm{y}_i, \widehat{\mathrm{y}}_i)$.

Batch Selection + minibatch SGD = Bilevel Optimization Solver

Consider a scenario where one is minimizing the overall empirical risk $L(w)$ via minibatch SGD.

The minibatch SGD algorithm picks b of the m indices uniformly at random, say $j_1, j_2, \ldots, j_b$, and updates its iterate with $\frac{1}{b}\sum_{i=1}^{b} \nabla\ell(y_{j_i}, \widehat{y}_{j_i})$, called a batch gradient. Note that a batch gradient is an unbiased estimate of the true gradient $\nabla L(w)$.

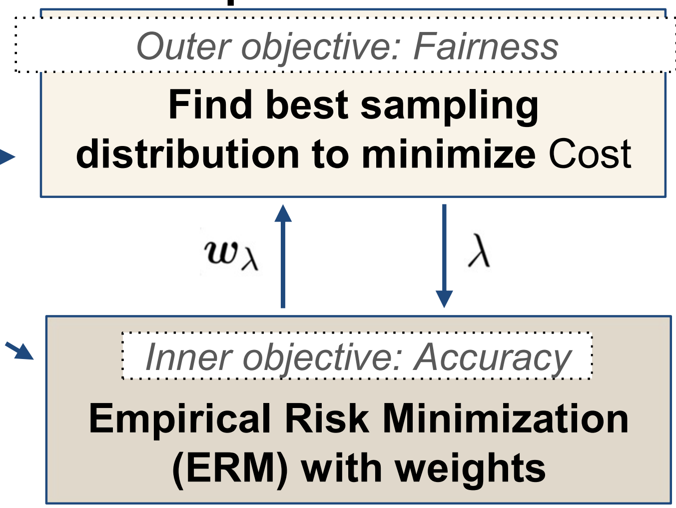
Not **uniform distribution**? if we draw train example i with probability p_i for all i such that $\sum p_i = 1$, the batch gradient is an unbiased estimate of $L'(w) = \sum_i p_i \ell(y_i, \widehat{y}_i)$

Batch Selection + minibatch SGD = Bilevel Optimization Solver

$$\min_{\boldsymbol{p}} \mathrm{Cost}(\boldsymbol{w_p})$$

$$\boldsymbol{w_p} = \arg\min_{\boldsymbol{w}} \sum_i p_i \ell(\mathrm{y}_i, \hat{\mathrm{y}}_i),$$

$$\boldsymbol{p} := (p_1, p_2, \ldots, p_m)$$



Algorithm 1: Bilevel optimization with `MinibatchSGD`

Minibatch sampling distribution $\leftarrow$ Uniform sampling
for *each epoch* **do**
  Draw minibatches according to minibatch sampling distribution
  for *each minibatch* **do**
    $\boldsymbol{w} \leftarrow$ `MinibatchSGD`$(\boldsymbol{w}$, each minibatch$)$
  Update minibatch sampling distribution

- How to design the cost function to capture a desired fairness criterion?
- How to design an update rule for the outer optimizer?

Equalized Odds $P(\hat{Y}=1|Z=0,Y=A)=P(\hat{Y}=1|Z=1,Y=A)$

Equalized odds requires the prediction to be independent from the sensitive attribute conditional on the true label, i.e., $L_{0,0}(\boldsymbol{w}) = L_{0,1}(\boldsymbol{w})$ and $L_{1,0}(\boldsymbol{w}) = L_{1,1}(\boldsymbol{w})$.

$L_{y,z}(\boldsymbol{w})$: the empirical risk aggregated over samples subject to $\mathrm{y} = y$ and $\mathrm{z} = z$;

To mitigate these disparities, we adjust (i) the sampling probability between $L_{0,0}(\boldsymbol{w})$ and $L_{0,1}(\boldsymbol{w})$ and (ii) the sampling probability between $L_{1,0}(\boldsymbol{w})$ and $L_{1,1}(\boldsymbol{w})$.

$$\min_{\boldsymbol{\lambda}\in[0,\frac{m_{0,\star}}{m}]\times[0,\frac{m_{1,\star}}{m}]} \max\{|L_{0,0}(\boldsymbol{w}_{\boldsymbol{\lambda}}) - L_{0,1}(\boldsymbol{w}_{\boldsymbol{\lambda}})|, |L_{1,0}(\boldsymbol{w}_{\boldsymbol{\lambda}}) - L_{1,1}(\boldsymbol{w}_{\boldsymbol{\lambda}})|\},$$

$$\boldsymbol{w}_{\boldsymbol{\lambda}} = \arg\min_{\boldsymbol{w}} \lambda_1 L_{0,0}(\boldsymbol{w}) + (\tfrac{m_{0,\star}}{m} - \lambda_1) L_{0,1}(\boldsymbol{w}) + \lambda_2 L_{1,0}(\boldsymbol{w}) + (\tfrac{m_{1,\star}}{m} - \lambda_2) L_{1,1}(\boldsymbol{w})$$

Demographic parity $P(\hat{Y}=1|Z=0)=P(\hat{Y}=1|Z=1)$

Demographic parity is satisfied if two sensitive groups have equal positive prediction rates, i.e., $L_{0,0}(\boldsymbol{w}) = L_{0,1}(\boldsymbol{w})$ and $L_{1,0}(\boldsymbol{w}) = L_{1,1}(\boldsymbol{w})$.

To satisfy this sufficient condition, we now adjust (i) the the sampling probability between $L_{0,0}(\boldsymbol{w})$ and $L_{1,0}(\boldsymbol{w})$ and (ii) the the sampling probability between $L_{0,1}(\boldsymbol{w})$ and $L_{1,1}(\boldsymbol{w})$.

$$\min_{\boldsymbol{\lambda}\in[0,\frac{m_{\star,0}}{m}]\times[0,\frac{m_{\star,1}}{m}]} \max\{|L_{0,0}(\boldsymbol{w}_{\boldsymbol{\lambda}}) - L_{1,0}(\boldsymbol{w}_{\boldsymbol{\lambda}})|, |L_{0,1}(\boldsymbol{w}_{\boldsymbol{\lambda}}) - L_{1,1}(\boldsymbol{w}_{\boldsymbol{\lambda}})|\},$$

$$\boldsymbol{w}_{\boldsymbol{\lambda}} = \arg\min_{\boldsymbol{w}} \lambda_1 L_{0,0}(\boldsymbol{w}) + (\tfrac{m_{\star,0}}{m} - \lambda_1) L_{1,0}(\boldsymbol{w}) + \lambda_2 L_{0,1}(\boldsymbol{w}) + (\tfrac{m_{\star,1}}{m} - \lambda_2) L_{1,1}(\boldsymbol{w})$$

Bi-level Optimization for FairBatch

$$\min_{\lambda \in [0, c_1]} F(\lambda) = |f_1(\boldsymbol{w}_\lambda) - g_1(\boldsymbol{w}_\lambda)|$$

$$\boldsymbol{w}_\lambda = \arg\min_{\boldsymbol{w}} \lambda f_1(\boldsymbol{w}) + (c_1 - \lambda) g_1(\boldsymbol{w}) + h(\boldsymbol{w})$$

Outer objective: Fairness

Find best λ value to minimize fairness cost

$\boldsymbol{w}_\lambda$ λ

Inner objective: Accuracy

Empirical Risk Minimization (ERM) with weights

Based on the quasi-convexity* of F(·), we design the signed gradient-based optimization algorithm:

$$\forall t \in \{0, 1, \ldots\} : \lambda^{(t+1)} = \lambda^{(t)} - \alpha \cdot \mathrm{sign}(g_1(\boldsymbol{w}_\lambda) - f_1(\boldsymbol{w}_\lambda))$$

* $F(t\lambda + (1-t)\lambda') \leq \max\{F(\lambda), F(\lambda')\}$ *for all* $t \in [0,1]$

Sample Code for Model Training

```
loader = DataLoader(training_data, sampler = sampler)

for epoch in range(epochs):
  for i, data in enumerate(loader):
    # get the inputs; data is a list of [inputs, labels]
        inputs, labels = data
        optimizer.zero_grad()
        outputs = model(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

return model
```

Pre- and In-Processing Approaches

Pre-processing: Fix training data

```
loader = DataLoader(training_data, sampler = sampler)

for epoch in range(epochs):
  for i, data in enumerate(loader):
    # get the inputs; data is a list of [inputs, labels]
        inputs, labels = data
        optimizer.zero_grad()
        outputs = model(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

return model
```

In-processing:
Fix training algorithm

Pre- & In-processing approaches require significant non-trivial changes in either data generation or algorithmic design

Simple Employment of FairBatch

```
fairsampler = FairBatch(model, target_fairness, ...)
loader = DataLoader(training_data, sampler = fairsampler)

for epoch in range(epochs):
  for i, data in enumerate(loader):
    # get the inputs; data is a list of [inputs, labels]
        inputs, labels = data
        optimizer.zero_grad()
        outputs = model(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()

return model
```

Experimental Settings

Datasets: COMPAS, AdultCensus (GENDER as the sensitive attribute)

Model: logistic regression

Measuring Fairness: equalized odds (ED) and demographic parity (DP)

$$\textbf{ED disparity} = \max_{z\in\mathbb{Z},y\in\mathbb{Y},\widehat{y}\in\widehat{\mathbb{Y}}} |\Pr(\widehat{\mathrm{y}} = \widehat{y}|\mathrm{z} = z, \mathrm{y} = y) - \Pr(\widehat{\mathrm{y}} = \widehat{y}|\mathrm{y} = y)|$$

$$\textbf{DP disparity} = \max_{z\in\mathbb{Z}} |\Pr(\widehat{\mathrm{y}} = 1|\mathrm{z} = z) - \Pr(\widehat{\mathrm{y}} = 1)|.$$

Experimental Results (*ED disparity*)

FairBatch achieves **fair** and **accurate** results **efficiently**

		COMPAS			AdultCensus		
		Accuracy	Unfairness	Epochs	Accuracy	Unfairness	Epochs
Vanilla	Logistic regression	.681	.239	300	.845	.054	300
Pre-processing	Reweighing [1]	.685	.137	300	.835	.134	100
	Label bias correction [2]	.673	.031	3900	.841	**.011**	6300
In-processing	Adversarial debiasing [3]	.683	.067	300	.841	<u>.016</u>	400
	AdaBoost-based fair training [4]	.664	**.018**	9600	.844	.038	9000
Batch Selection	**FairBatch**	.681	<u>.022</u>	100	.844	**.011**	400

[1] Kamiran and Calders, 2011 [2] Jiang and Nachum, 2020
[3] Zhang et al., 2018 [4] Iosifidis and Ntoutsi, 2019

Experimental Results (*ED disparity*)

FairBatch achieves **fair** and **accurate** results **efficiently**

		COMPAS			AdultCensus		
		Accuracy	Unfairness	Epochs	Accuracy	Unfairness	Epochs
Vanilla	Logistic regression	.681	.239	300	.845	.054	300
Pre-processing	Reweighing [1]	.685	.137	300	.835	.134	100
	Label bias correction [2]	.673	.031	3900	.841	**.011**	6300
In-processing	Adversarial debiasing [3]	.683	.067	300	.841	.016	400
	AdaBoost-based fair training [4]	.664	**.018**	9600	.844	.038	9000
Batch Selection	**FairBatch**	.681	.022	100	.844	**.011**	400

Fair, but slow

Fair, accurate, and fast

[1] Kamiran and Calders, 2011 [2] Jiang and Nachum, 2020
[3] Zhang et al., 2018 [4] Iosifidis and Ntoutsi, 2019

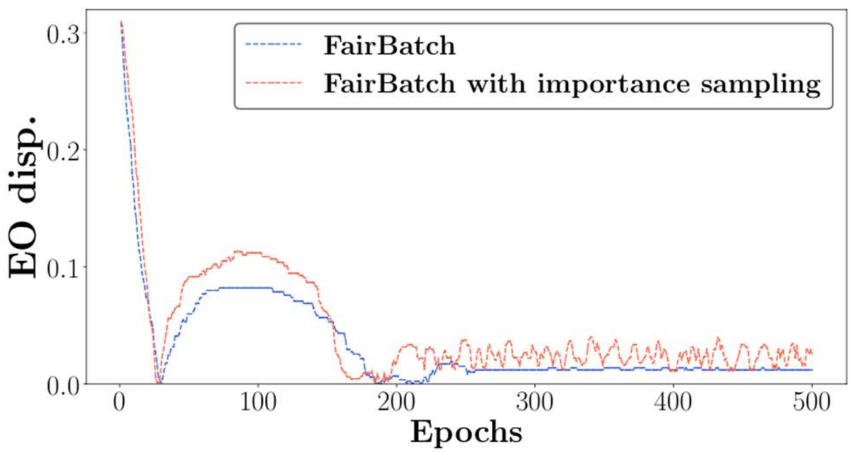
Experimental Results (*demographic parity*)

		COMPAS			AdultCensus		
		Accuracy	Unfairness	Epochs	Accuracy	Unfairness	Epochs
Vanilla	Logistic regression	.681	.192	300	.845	.125	300
Pre-processing	Reweighing [1]	.685	.103	300	.835	.052	300
	Label bias correction [2]	.671	**.032**	7900	.815	<u>.011</u>	12600
In-processing	Adversarial debiasing [3]	.683	.054	550	.815	.018	400
	AdaBoost-based fair training [4]	.642	<u>.033</u>	6300	.825	.040	27000
Batch Selection	**FairBatch**	.681	.036	300	.823	**.010**	600

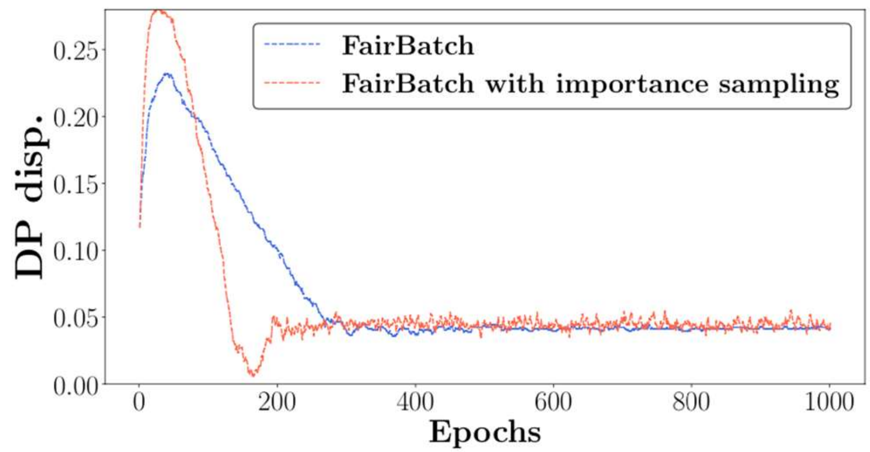
[1] Kamiran and Calders, 2011 [2] Jiang and Nachum, 2020
[3] Zhang et al., 2018 [4] Iosifidis and Ntoutsi, 2019

Experimental Results

FairBatch: Compatibility with existing batch selection approaches that use importance sampling for faster convergence in training



(a) EO disparity curve of FairBatch.



(b) DP disparity curve of FairBatch.

Conclusion

- FairBatch improves model fairness and accuracy efficiently with a one-line change of code
- Idea: Adaptively selects batch sizes to improve fairness using bi-level optimization
- Also gracefully merges with existing batch selection techniques used for faster convergence