AN ANALYSIS OF AI ALGORITHMS ON

Recidivism

Jackson Kaunismaa, Julia Wang, Ryan Chen, Vishnu Akundi

1

Why

The impact of AI misclassification

The Issue in Recidivism Algos

- Recidivism algorithms are used in the real world by probation departments
- Defendants classified as higher risk more likely detained
- Northpointe AI assessment tool
  - race impacted results even when not an input

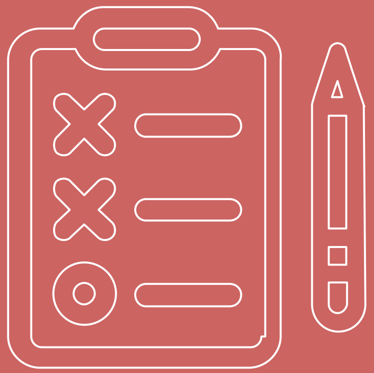
Unethical treatment of individuals of color

Our Purpose

Build an AI model classifying risk of recidivism with minimal bias

Metrics

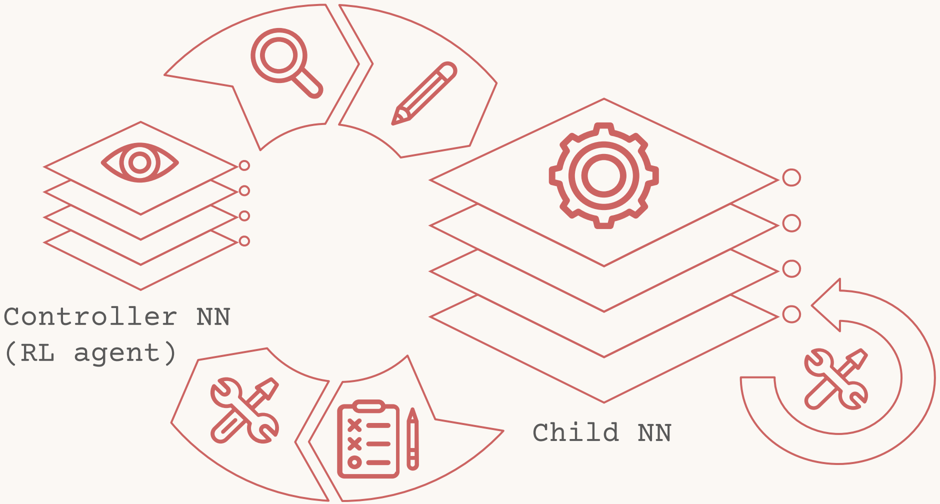
- Many fairness metrics are useful only to guide model design
- Cannot be directly incorporated into a loss function due to non-differentiability (Kusner et al., 2017).
- Therefore, use Neural Architecture Search (NAS) with the fairness metric as a reward signal



2

Our Model

A feedforward neural net with NAS



Data processing

- COMPAS dataset
- Trained on features:
  - Sex, age, race, time in jail, number of juvenile felonies, juvenile misdemeanors, and other juvenile counts, prior convictions, charge degree
- Temporally split data, 80:10:10 training-validation-test
- We chose to look at counterfactual fairness, as an example of one of the non-differentiable fairness metrics that would otherwise be hard to incorporate in an end-to-end fashion

Counterfactual Fairness

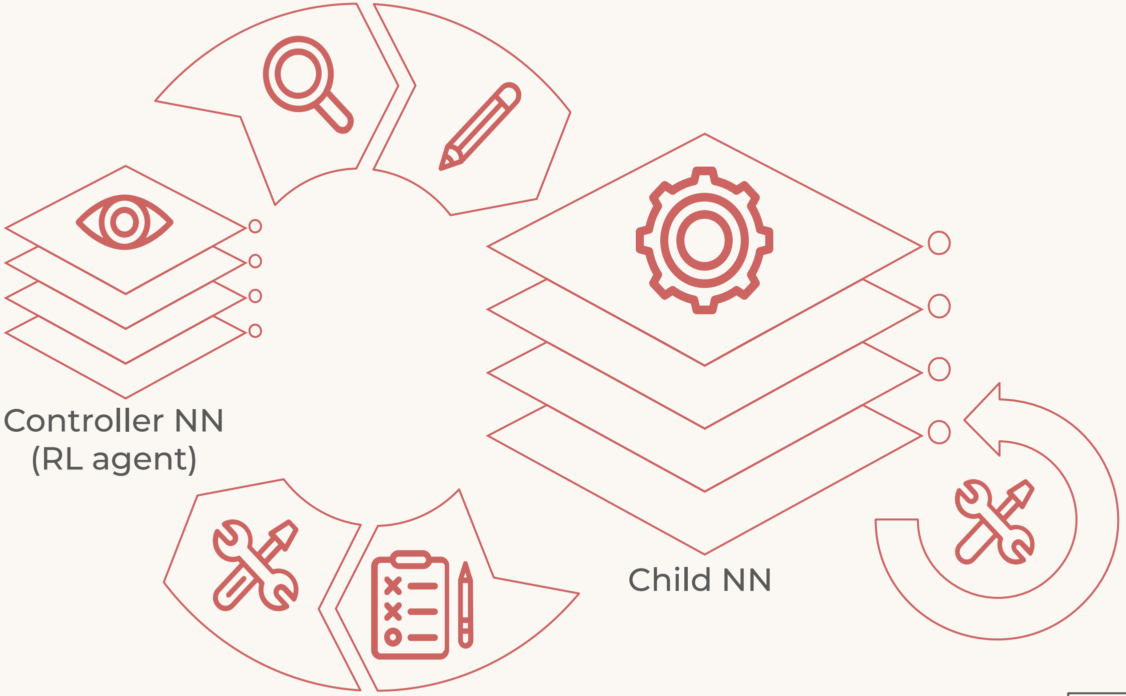
a model is fair if the real world prediction for an individual or demographic is the same as in the counterfactual world where they're from a different demographic

```
def counter_factual(self, x, sensitive_trait, transforms):
  classif_probs = []
  for transform in transforms:
    classifs = self.classify(*transform(x, sensitive_trait))
    classif_probs.append((classifs.sum()/classifs.shape[0]).detach().cpu())
  return np.std(classif_probs)
```

Neural Architecture Search (NAS)

The main predictions for 2-year recidivism are made by the Child NN

The Controller NN uses the reward and counterfactual fairness metric provided by the Child NN validation to adjust the Child NN's sensitive characteristic thresholds and hidden layer sizes.



Hyperparameter Tuning

Methodology

Our method uses an additional recurrent controller network alongside the primary prediction neural net (child), in line with Zoph et al. (2016).

```
def forward(self, x):
  # input is all zeros vector
  hidden = torch.zeros(self.hidden_size)
  outputs = []
  hiddens = []
  for _ in range(self.num_outputs):
    hidden, x = self.run_once(hidden, x)
    outputs.append(x.clone())
    hiddens.append(hidden.clone())
  return outputs, hiddens
```

```
for j in range(3):
  threshs, hiddens, init_gen = controller.select_params()
  child = ChildNetwork(15, 1, hiddens, ACT_FUNCS, torch.tensor(thr
  chld_optim = torch.optim.Adam(child.parameters())
  final_va_acc, final_va_counter_fact = train(child, chld_optim, t

  reward = final_va_acc + final_va_counter_fact*3.0
```

Hyperparameter Tuning

Execution

The control network (RL agent) is able to learn several critical hyperparameters through the feedback from the child network.

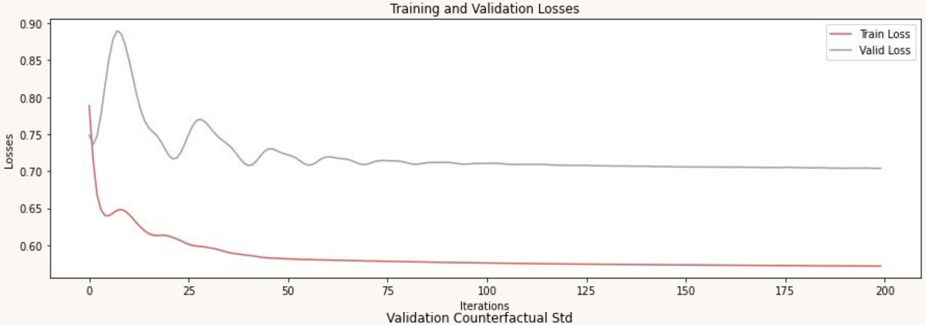
```
for _ in range(self.num_outputs):
  hidden, x = self.run_once(hidden, x_inpt)
  hidden = hidden.clone().detach()
  x_inpt = x.clone().detach()
  objective = -torch.log(x.max()) * (reward - prev_reward)
  objective.backward()
  optim.step()
  optim.zero_grad()
```

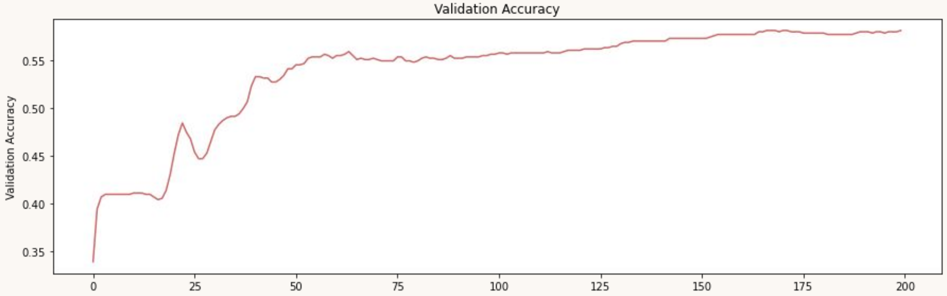
3

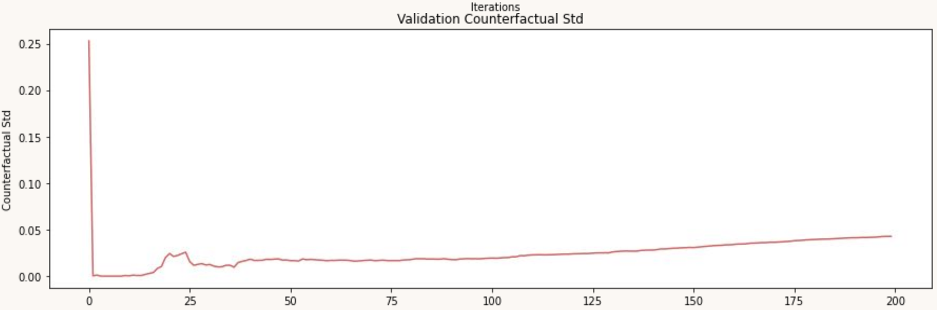
Results

Using Neural Architecture Search to optimize for Fairness.

Naive Classifier



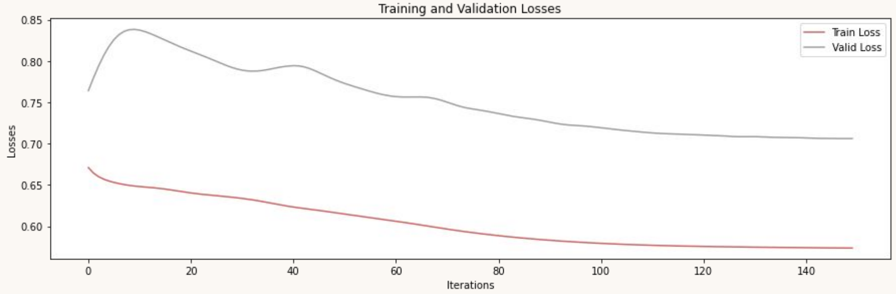


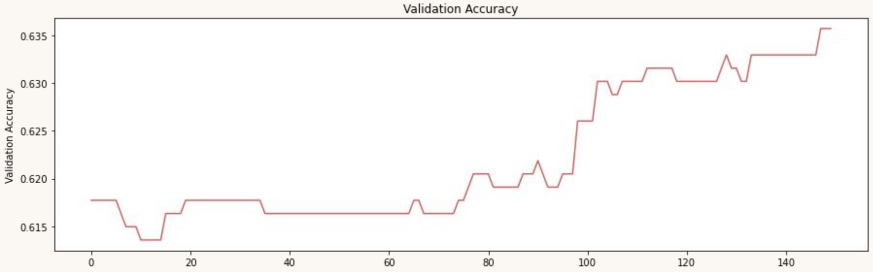


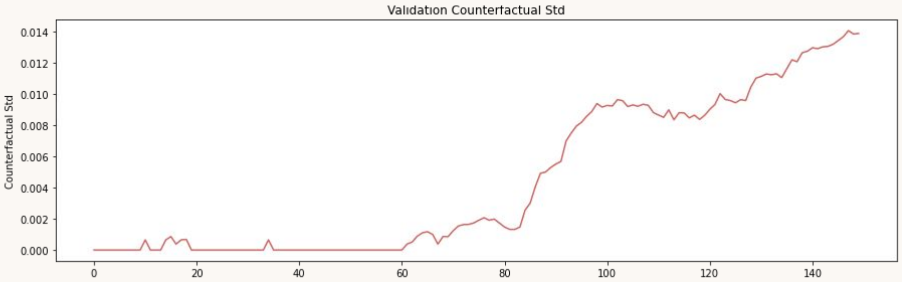
Accuracy: 61.9%

CF Fairness: 0.043

Result of Neural Architecture Search







Accuracy: 63.6%

CF Fairness: 0.013

Key Results

Counterfactual Fairness

3.5x reduction in counterfactual fairness*

Accuracy

Does not harm accuracy significantly

Fairness

Developed general method for incorporating non-differentiable fairness metrics into model

*- As measured by the standard deviation of positive classification rates between demographic categories

Next Steps

- Generalizing Results
  - Reproduce existing results
  - Using other types of Fairness Metrics
  - Using other Datasets
  - Different types of Child Networks
- Refine calculations for Controller Network’s gradients.
- Investigate counterfactual standard deviation calculation validity

Thanks for listening

References

https://arxiv.org/pdf/1703.06856.pdf - Kusner (counterfacutal fairness/non-differentiability)
https://arxiv.org/abs/1611.01578 - Zoph (NAS)