This document is a summary of the *Reliable and Trustworthy Artificial Intelligence* course at ETH Zürich. This summary was created during the autumn semester of 2023. Due to updates to the syllabus content, some material may no longer be relevant for future versions of the course. I do not guarantee correctness or completeness, nor is this document endorsed by the lecturers. This draft cheatsheet exceeds the allowed number of pages for the exam, so make sure to cut it down to size. For the full LaTeX source code, visit `github.com/Jovvik/eth-cheatsheets`.

1 Adversarial Attacks

Targeted FGSM: $x' = x - \eta$, $\eta = \epsilon \cdot sign(\nabla_x loss_t(x))$
Untarg. FGSM: $x' = x + \eta$, $\eta = \epsilon \cdot sign(\nabla_x loss_s(x))$
Guarantees $\eta \in [-\epsilon, \epsilon]$, η not minimized
Carlini & Wagner: Find targeted adv. sample $x' = x + \eta$ *and* minimize $||\eta||_p$ via minimizing $||\eta||_p + c \cdot \text{obj}_t(x')$, where obj_t is s.t. $\text{obj}_t(x') \leq 0 \Rightarrow f(x') = t$, e.g. $CE(x', t) - 1; \max(0, 0.5 - p_f(x')_t)$
Prior e.g. $x + \eta \in [0,1]$: use specialized optimizer (LBFGS-B) or PGD. Optimizing $||\eta||_\infty$ is hard, use $\text{ReLU}(\sum |\eta_i| - \tau)$, lower τ gradually.
PGD: Repeat FGSM with ϵ_{step} and proj. to $x \pm \epsilon$.

2 Adversarial Defenses

$\min_\theta \mathbb{E}_{(x,y)\sim D}[\max_{x' \in S(x)} L(\theta, x', y)]$
usually $S(x) = \mathbb{B}^\infty_\epsilon$, $\mathbb{E} \approx$ empirical risk.
PGD Defense algorithm: Run PGD on every batch and use $\nabla_\theta \mathcal{L}(x_{\text{adv}})$ for backprop.
TRADES defense:
$\min_\theta \mathbb{E}_{(x,y)\sim D}[L(\theta, x, y) + \lambda \max_{x' \in \mathbb{B}_\epsilon(x)} L(\theta, x', y)]$

3 Certification of Neural Networks

Given NN N, precond. ϕ, postcond. ψ prove: $\forall i \;\; i \vDash \phi \Rightarrow N(i) \vDash \psi$ or return a violation.

3.1 Complete Methods (always return result)

Encode NN as MILP instance. Doesn't scale well.
- Affine: $y = Wx + b$ is a direct MILP constraint. $Wx + b \leq y \leq Wx + b$.
- ReLU(x): $y \leq x - l_x \cdot (1 - a)$, $y \geq x$, $y \leq u_x \cdot a$, $y \geq 0$, $a \in \{0, 1\}$, for box bound $x \in [l, u]$.
  - $a = 0$: $y = 0, x \in [l, 0]$
  - $a = 1$: $y = x, y \in [0, u]$

To check an encoding for f, plot constraint regions for all cases of int. variables. They should match plot of f. Can't use $a \cdot x$.
$\phi = \mathbb{B}^\infty_\epsilon(x)$: $x_i - \epsilon \leq x'_i \leq x_i + \epsilon, \forall i$
precomp. Box bounds: $l_i \leq x^p_i \leq u_i$
$\psi = o_0 > o_1$: MILP objective $\min o_0 - o_1$.

3.2 Incomplete Methods (may abstain)

Over-approximate ϕ using relaxation, then push approximation through NN via bound propagation.
Box($O(n^2 L)$): Bounds are l_∞ balls. $[a,b] +^\# [c,d] = [a+c, b+d]$, $-^\# [a,b] = [-b, -a]$; $\text{ReLU}^\#[a,b] = [\text{ReLU}(a), \text{ReLU}(b)]$; $\lambda \cdot^\# [a,b] = [\lambda a, \lambda b]$ $(\lambda \geq 0)$
DeepPoly($O(n^3 L^2)$): For each x_i keep constraints: interval $l_i \leq x_i, x_i \leq u_i$; relational $a^\leq_i \leq x_i, x_i \leq a^\geq_i$ where $a^\leq_i, a^\geq_i$ are of the form $\sum_j w_j \cdot x_j + v$
- $x_j = \text{ReLU}^\#(x_i)$: interval constr. $x_i \in [l_i, u_i]$:
$u_i \leq 0$: $a^\leq_j = a^\geq_j = 0, l_j = u_j = 0$;
$l_i \geq 0$: $a^\leq_j = a^\geq_j = x_i, l_j = l_i, u_j = u_i$;
$l_i < 0, u_i > 0$: $\lambda \coloneqq u_i/(u_i - l_i), x_j \leq \lambda(x_i - l_i), \alpha \in [0,1], \alpha x_i \leq x_j, l_j = 0, u_j = u_i$.
Min area: if $u \leq -l$, $\alpha = 0$, otherwise 1.
When proving $y_2 > y_1$, add a layer that computes $y_2 - y_1$ and prove $l_{y_2 - y_1} > 0$.
Branch & Bound: Split ReLU based on $x_i \leq 0$, resulting bound is the worst of two cases. Naive split still covers extra space, need constraints. KKT: $(\max f(x) \mid g(x) \leq 0) \leq \max_x \min_\beta f(x) - \beta g(x)$
- $(\max_x \vec{a}\vec{x} + c \text{ s.t. } -x_i \leq 0) \leq \max_x \min_\beta \vec{a}\vec{x} + c + \beta x_i$ - $(\max_x \vec{a}\vec{x} + c \text{ s.t. } x_i \leq 0) \leq \max_x \min_\beta \vec{a}\vec{x} + c - \beta x_i$ Usually you use the weak duality after this. β is found by GD, and on each step you do full backsubstitution after the split, as the sign in front of symbolic variables can change when β changes.

4 Certified Defenses

Produces models that are easier to certify.

4.1 DiffAI

PGD: $\min \mathbb{E}_{(x,y)\sim D}[\max_{z \in \gamma(NN^\#(S(x)))} L(\theta, z, y)]$
Can use any abstract transformer (Box, DeepPoly).
To find max loss, use abstract loss $L^\#(\vec{z}, y)$, where y = target label, $\vec{z}$ = vector of logits:
- $L(z, y) = max_{q \neq y}(z_q - z_y)$: Compute $d_c = z_c - z_y$ $\forall c \in C$, where z_c the abstract logit shape of class i. Then compute box bounds of d_c and compute max upper bound: $\max_{c \in C}(\max(box(d_c)))$
- $L(z, y) = CE(z, y)$: Compute box bounds $[l_c, u_c]$ of z_c. $\forall c \in C$ pick u_c if $c \neq y$, pick l_c if $c = y$, hence $v = [u_1, .., l_c, .., u_{|C|}]$. Compute $CE(\text{softmax}(v), y)$.

4.2 COLT

COLT: Run relaxation up to some layer: $S' = NN^\#_{1\ldots i}(S(x))$, then run PGD on the region to train layers $i + 1 \ldots n$. For PGD we need to project back to S', which is not efficient for DeepPoly.

5 Randomized Smoothing for Robustness

Given any classifier f, make a smoothed classifier $g(x) \coloneqq \arg\max_{c_A \in Y} \mathbb{P}_\varepsilon(f(x + \varepsilon) = c_A)$, where $\varepsilon \sim \mathcal{N}(0, \sigma I)$, $p_A(x)$ is the probability under argmax.
If $\exists \underline{p_{A,x}}, \overline{p_{B,x}} \in [0,1]$ s.t. $p_A(x) \geq \underline{p_{A,x}} \geq \overline{p_{B,x}} \geq \max_{B \neq A} p_B(x)$, then $g(x + \delta) = c_A \quad \forall ||\delta||_2 < R_x$ aka certification radius $= \delta/2(\Phi^{-1}(\underline{p_{A,x}}) - \Phi^{-1}(\overline{p_{B,x}}))$.

Calculating $p_{A,x}, p_{B,x}$ directly is hard, so we use bounds. Calculating $\overline{p_{B,x}}$ is also hard, so let's assume $\underline{p_{A,x}} > 0.5$, then $\overline{p_{B,x}} = 1 - \underline{p_{A,x}}$.

```
ĉ_A ← guess_top_class(f, σ, x, n_0)
p_A ← lower_bound_p(ĉ_A, f, σ, x, n, α)
if p_A > 0.5 then
    R ← σΦ^{-1}(p_A)
    return ĉ_A R
else
    return ABSTAIN
end if
```

Top class is estimated via Monte-Carlo. Lower bound is estimated by CLT, Chebyshev's inequality or binomial confidence bounds. The two function calls involve sampling, the samples should be separate, and $n \gg n_0$. If the algorithm returns ABSTAIN, one of the following is true:

- $\hat{c_A}$ is wrong, fixed by increasing n_0
- True $p_A \leq 0.5$, unfixable
- Lower bound is too low, fixed by increasing n

Inference:

```
ĉ_A, n_A, ĉ_B, n_B ← top_two_classes(f, σ, x, n)
return BinomPValue(n_A, n_A+n_B, =, 0.5) ≤ α
  ? ĉ_A : ABSTAIN
```

- NH: true p(success) of f returning $\hat{c}_A$ is 0.5
- BinomPValue returns p-value of null hypothesis, evaluated on n iid samples with i successes.
- Accept NH if p-value is $> \alpha$, reject otherwise.
- α small: often accept null hypothesis and ABSTAIN, but more confident in predictions.
- α large: more predictions but more mistakes.
- Returns wrong class with probability at most α

6 Privacy

Common attacks: Model Stealing, Model Extraction (representative inputs), Data Extraction (exact training samples), Membership Inference (find out if a sample was used for training).

Black-Box MI: Attacker trains many models on the same data distribution, some with entry x, some without. If logits are given, then attacker trains a classifier to distinguish between the two cases. If not, then do the same with robustness scores.

6.1 Federated Learning

FedSGD: Entities do training steps on minibatches x^k, y^k from private data $\mathcal{D}_k$ and return gradients $g_k \coloneqq \nabla_\theta \mathcal{L}(f_{\theta_t}(x^k), y^k)$, average on server and update the global model $\theta_{t+1} \coloneqq \theta_t - \gamma g_c$. But sent data still contains information about private data.
Honest but curious server: Server does not manipulate sent weights. For batch size 1 and piecewise linear activation functions, the server can learn the data exactly. For batch size $>$ 1 and some assumptions, a linear combination of some true inputs can be found. The general approach is: $\arg\min_{x^*} d(g_k, \nabla_\theta \mathcal{L}(f_{\theta_t}(x^*), y^*)) + \alpha_{\text{reg}} \cdot \mathcal{R}(x^*)$

- d is distance, typically l_1, l_2 or cosine.
- $\mathcal{R}$ is a prior based on domain-specific knowledge.
- Optimization is done via GD.
- y^* is recovered separately (out of scope).
- For each categorical feature create an N-dim. variable that gets put into x^* through softmax.

For tables, we can use entropy over many randomly initialized reconstructions as a prior, because correct cells are robust to random initializations.
FedAVG: Client runs E epochs of SGD, sends new weights to server. Final weights depend on order of batches, the server does not know it. Attack simulates training. Prior: the average of samples in one epoch is equal to that in another epoch.

6.2 Differential Privacy

MI protection is $\mathbb{P}(M(\mathcal{D}) \in S) \approx \mathbb{P}(M(\mathcal{D}) \in S)$
M is ε-DP if for all "neighboring" (a, a') and for any attack S $p(a) \coloneqq \mathbb{P}(M(a) \in S) \leq e^\varepsilon \mathbb{P}(M(a') \in S)$.
As $e^\varepsilon \approx 1 + \varepsilon$, $(1 - \varepsilon)p(a') \lesssim p(a) \lesssim (1 + \varepsilon)p(a')$.
By a theorem, $f(a) + \text{Lap}(0, \Delta_1/\varepsilon)$ is ε-DP, where $\Delta_p \coloneqq \max_{(a,a') \in \text{Neigh}} ||f(a) - f(a')||_p$.
M is ε, δ-DP iff $\mathbb{P}(M(a) \in S) \leq e^\varepsilon \mathbb{P}(M(a') \in S) + \delta \; \forall (a, a') \in \text{Neigh}, \forall S$. This allows absolute differences (not only relative). If $p(a') = 0$, $p(a) \neq 0$, no ε-DP mechanism exists, but ε, δ-DP might.
If output set is discrete, singleton attacks are enough. $f(a) + \mathcal{N}(0, \sigma^2 I)$ is ε, δ-DP, where $\sigma = \sqrt{2\log(1.25)/\delta} \cdot \Delta_2/\varepsilon$.
If M_1, M_2 are ε_1, δ_1-DP and ε_2, δ_2-DP, then (M_1, M_2) and $M_1 \circ M_2$ are $\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2$-DP

In particular, if f is a plain function (0, 0-DP), then $f \circ M$ is ε, δ-DP. If A_i has user data and M_i is $(\varepsilon_i, \delta_i)$-DP, $M_1(a_1) \dots M_k(a_k)$ is $(\max_i \varepsilon_i, \max_i \delta_i)$-DP.

DP-SGD: Project gradients for each point onto l_2-ball of size C and sum them up. Add $\mathcal{N}(0, \sigma^2 I)$ to the batch gradient, where $\sigma = \sqrt{2\log(1.25)/\delta} \cdot C/L/\varepsilon$ The resulting model is private, even against a white-box attacker with any number of queries. Clipping is necessary to bound the sensitivity of the gradient.

Privacy Amplification: Applying an (ε, δ)-DP mechanism on a random fraction $q = L/N$ subset yields a $(\tilde{q}\varepsilon, q\delta)$-DP mechanism, where $\tilde{q} \approx q$.

Due to clipping, sensitivity of the gradient for any point is C. If $T = 1$ and no subsampling is used, adding/removing a datapoint changes total gradient by at most C/L. Then by the gaussian mechanism the resulting model is ε, δ-DP. If subsampling is used, by privacy amplification, the model is $(\tilde{q}\varepsilon, q\delta)$-DP. If $T \neq 1$, by the composition theorem, the model is $(\tilde{q}T\varepsilon, qT\delta)$-DP. By out of scope theorems, this is $(O(q\varepsilon\sqrt{T \log \frac{1}{\delta}}), O(qT\delta))$ and $(O(q\varepsilon\sqrt{T}), \delta)$-DP.

PATE: Private Aggregation of Teacher Models

Split data into disjoint partitions and train a model for each. Agreggate models via noisy voting into a teacher, which labels public unlabeled data, on which we train the final model.

T are teachers, $n_j(x) \coloneqq |\{t(x) = j \mid t \in T\}|$. $\arg\max(n_j(x)) + \mathrm{Lap}(0, \sigma)$ is bad, better $\arg\max(n_j(x) + \mathrm{Lap}(0, 2/\varepsilon))$. $\Delta_1 = 2 \Rightarrow$ model is $(\varepsilon, 0)$-DP for one query. Labeling T data points yields $(\varepsilon T, 0)$-DP. But there are better bounds.

FedSGD/FedAVG with Noise: clip the gradients/weights and add noise.

DP is closely related to randomized smoothing. We add noise to data, then forward is ε-DP.

7 AI Regulation

Key issues: fairness, explainability, data minimization, unlearning (right to be forgotten), copyright.

8 Private synthetic data

Data is private, make DP synthetic proxy.

1. **Select** marginal queries we want to measure
2. **Measure** marginal queries using DP
3. **Generate** synthetic data

Marginal on $C \subseteq \mathcal{A}$ (attrs.) is a vector $\mu \in \mathbb{R}^{n_C}$, indexed by $t \in \Omega_C$, where $\Omega_C = \prod_{i \in C} \Omega_i$ and $n_C = |\Omega_C|$. Each entry μ_t is a count $\sum_{x \in D}[x_C = t]$. $M_C : \mathcal{D} \to \mathbb{R}^{n_C}, D \mapsto \mu$ computes the marginal. $\Delta_2(M_C) = 1$ because adding a row in a dataset can only change one element of the vector. 1-way marginals ($n_C = 1$) are histograms, 2-way marginals are heatmaps.

Mutual information of two variables X, Y is $\mathrm{I}(X, Y) = \sum_{x,y} \frac{p(x,y)}{p(x)p(y)}$. Chow-Liu algorithm makes a complete graph of features, edge weigths $\mathrm{I}(X, Y)$. Find MST, the optimal 2nd-order approximation. Generate by sampling from MST, each node is conditioned on its parent, i.e. $p(F_1 = f_1, F_2 = f_2, F_3 = f_3) = p(F_1 = f_1)p(F_2 = f_2 \mid F_1 = f_1)p(F_3 = f_3 \mid F_1 = f_1)$, if F_1 is parent of F_2 and F_3.

Add DP, i.e. add noise to every step of the algorithm. MST is done with the exponential mechanism, marginals are measured with Gaussian noise.

ProgSyn: allows to specify constraints.

- Sample random noise $z \sim \mathcal{N}(0, I_p)$
- Pass z through a generative model g_θ
- Get synthetic dataset $g_\theta(z)$
- Adapt θ to make $g_\theta(z)$ close to original X
- Fine-tune g_θ to make $g_\theta(z)$ satisfy constraints

9 Logic and Deep Learning (DL2)

9.1 Querying Neural Networks

Use standard logic ($\forall, \exists, \wedge, \vee, f : \mathbb{R}^m \to \mathbb{R}^n, ..$) and high-level queries to impose constraints.

$(class(NN(i)) = 9) = \bigwedge_{j=1, j\neq 9}^{k} NN(i)[j] < NN(i)[9]$

Use translation T of logical formulas into differentiable loss function $T(\phi)$ to be solved with gradient-based optimization to minimize $T(\phi)$. Regular SAT solvers can't handle non-small NNs.

Theorem: $\forall x, T(\phi)(x) = 0 \iff x \vDash \phi$

Logical Term	Loss
$t_1 \leq t_2$	$\max(0, t_1 - t_2)$
$t_1 \neq t_2$	$[t_1 = t_2]$
$t_1 = t_2$	$T(t_1 \leq t_2 \wedge t_2 \leq t_1)$
$t_1 < t_2$	$T(t_1 \leq t_2 \wedge t_1 \neq t_2)$
$\phi \vee \psi$	$T(\phi) \cdot T(\psi)$
$\phi \wedge \psi$	$T(\phi) + T(\psi)$

By construction $T(\phi)(x) \geq 0, \forall x, \phi$. Negation can be implemented by using de Morgan's laws.

Box constraints: hard to enforce in GD. Use L-BFGS-B and give box constraints to optimizer.

9.2 Training NN with Background Knowledge

Enforce logical property ϕ when training NN.

Problem statement: find θ that maximizes the expected value of property $\mathbb{E}_{s\sim D}[\forall z.\phi(z, s, \theta)]$. BUT: Universal quantifiers are difficult.

Reformulation: get the worst violation of ϕ and minimize its effect, i.e. $\mathbb{E}_{s\sim D}[\max_z \neg\phi(z, s, \theta)]$.

Reform. 2: minimize $\mathbb{E}_{s\sim D}[T(\phi)(bz, s, \theta)]$, where $bz = \mathrm{argmin}(T(\neg\phi)(z, s, \theta))$. This is an adv. attack.

$\exists$ different bz which minim. $T(\neg\phi)$ which can produce different $T(\phi)$. $bz \neq$ worst example.

Restrict z to a convex set with efficient projs., i.e. L_∞-balls. Remove the constraint from ϕ that restricts z on the convex set and do PGD while projecting z onto the convex set.

10 Fairness

A mapping $M : \mathcal{X} \to \Delta(\mathcal{Y})$ is (D, d)-**Lipschitz**, if for every $x_1, x_2 \in \mathcal{X}$ $D(M(x_1), M(x_2)) \leq d(x_1, x_2)$. If M is a model, it's **individually fair** wrt. D and d. d is a distance in feature space, D is a metric on probability distributions. Choosing metrics is hard.

Lemma: For $h : \mathbb{R}^d \to [0, 1]$, $x \mapsto \Phi^{-1}(\mathbb{E}_{\varepsilon\sim\mathcal{N}(0,I)}[h(x + \varepsilon)])$ is 1-Lipschitz in x.

Let $L \in \mathbb{R}$ be s.t. $D(M(x), M(x')) \leq Ld(x, x')$ (smaller value is stronger). Let $d(x, x') \coloneqq (x - x')^\top S(x - x')$, where S is a symmetric positive definite covariance matrix. Let $D(M(x), M(x')) \coloneqq [M(x) \neq M(x')]$. Then the Lipschitz property is equivalent to $\forall\delta \in \mathbb{B}_S(0, 1/L)$ $M(x) = M(x + \delta)$, where $||x||_S \coloneqq \sqrt{x^\top Sx}$. We have reformulated individual fairness as robustness.

10.1 Fair Representation Learning

FRL is often more efficient (reuse fair data) and simplifies audits. But it has less precise control of the fairness/performance tradeoff, is susceptible to adv. attacks by the consumer, can be expensive and provides no certification.

10.2 Learning Certified Individually Fair Representations

Keep pros of FRL, but also allow the regulator to certify the fairness of the E2E model and allows to define D and d via logical constraints that are accepted by MILP and DL2. Example: $d(x, x') = \bigwedge_{i\in\mathrm{Cat}\setminus\{\mathrm{race},\mathrm{gender}\}}(x_i = x_i') \bigwedge_{j\in\mathrm{Num}} |x_j - x_j'| \leq \alpha$. Logic captures cat. features exactly, norms don't.

Let $S_d(x)$ denote the set of all points similar to x and assume $D(M(x), M(x')) = [M(x) \neq M(x')]$.

The encoder $f_\theta : \mathbb{R}^n \to \mathbb{R}^k$ is trained using DL2 s.t. $\forall x' \in S_d(x)$ $||f_\theta(x) - f_\theta(x')||_\infty \leq \delta$. $S_d(x)$ is a complicated set, which we bound by a box in latent space. The producer encodes $S_d(x)$ and f_θ as MILP to compute ε s.t. $f_\theta(S_d(x)) \subseteq \{z' \mid ||f_\theta(x) - z'||_\infty \leq \varepsilon\}$, which gives the consumer a simple robustness problem.

Train encoder using 9.2 with classifier to keep latent space useful. Train decoder via 5.

10.3 Latent Space Smoothing for individually Fair Representations

Use semantic feature space from a good gen. model encoder for similarity formulas for images etc.

Center smoothing produces a bound on the radius of the ball in latent space. The E2E model is individually fair with probability $1 - \alpha_{\mathrm{rs}} - \alpha_{\mathrm{cs}}$.

10.4 Group Fairness

Demographic parity: $\mathbb{P}(\hat{Y} = 1 \mid G = 0) = \mathbb{P}(\hat{Y} = 1 \mid G = 1)$, where G is a group feature.

Equal opportunity: $\mathbb{P}(\hat{Y} = 1 \mid Y = 1, G = 0) = \mathbb{P}(\hat{Y} = 1 \mid Y = 1, G = 1)$

Equalized odds: Equal opportunity and $\mathbb{P}(\hat{Y} = 1 \mid Y = 0, G = 0) = \mathbb{P}(\hat{Y} = 1 \mid Y = 0, G = 1)$

Example of **postprocessing**: for a binary classifier with output probability $h(x)$. Use separate thresholds for each group, tuned to achieve group fairness.

Example of **in-training**: add relaxed fairness constraints that are solved with DL2, i.e. $-\varepsilon \leq \mathbb{P}(\hat{Y} = 1 \mid s = 0) - \mathbb{P}(\hat{Y} = 1 \mid s = 1) \leq \varepsilon$

Preprocessing: FRL. Notation: data $(x, s) \in \mathbb{R}^d \times \{0, 1\}$, encoder $f : \mathbb{R}^d \times \{0, 1\} \to \mathbb{R}^{d'}$, $z = f(x, s)$, classifier $g : \mathbb{R}^{d'} \to \{0, 1\}$, adversary $h : \mathbb{R}^{d'} \to \{0, 1\}$ is a classifier that tries to predict the sensitive attribute from data in the latent space, $Z_i \coloneqq \{z \mid s = i\}$, $p_i(z) \coloneqq \mathbb{P}(z \mid s = i)$.

LAFTR: jointly train f, g and h. No guarantees. $\min_{f,g}\max_h(\mathcal{L}_{clf}(f(x, s), g) - \gamma\mathcal{L}_{adv}(f(x, s), h))$

Use adversary to add guarantees by computing an upper bound on unfairness of any g. Convert hard constraint (DP, EO) into a soft measure, e.g. for demographic parity: $\Delta_{Z_0,Z_1}(g) \coloneqq |\mathbb{E}_{z\sim Z_0} g(z) - \mathbb{E}_{z\sim Z_1} g(z)|$, lower is better. Balanced accuracy is $BA_{Z_0,Z_1}(h) = \frac{1}{2}(\mathbb{E}_{z\sim Z_0}(1 - h(z)) + \mathbb{E}_{z\sim Z_1} h(z)) = \frac{1}{2}\int_z (p_0(z)(1 - h(z)) + p_1(z)h(z))$, h chooses p_0

or p_1. The optimal adversary is $h^*(z) \coloneqq [p_1(z) \geq p_0(z)]$. Theorem: $\Delta_{Z_0,Z_1}(g) \leq 2 \cdot BA_{Z_0,Z_1}(h^*) - 1$. We can't find neither BA nor h^* exactly.

Fair Normalizing Flows: sample x from a known distribution q, apply an invertible encoder $z = f(x)$, find density of the new distribution by $\log p(z) = \log q(f^{-1}(z)) + \log|\det \frac{\partial f^{-1}(z)}{\partial z}|$. Learn normalizing flows f_0 and f_1 as encoders for Z_0 and Z_1. This lets us find $p_0(z)$ and $p_1(z)$, given $q_0(x), q_1(x)$. They can be estimated with density estimation, e.g. Gaussian Mixture Model. Given $p_0(z), p_1(z)$, we estimate an UB of BA with probability $1-\varepsilon$ by Hoeffding's inequality, and then apply the theorem for UB of Δ.

For good bounds, need low accuracy of $h^* \implies$ low dist. between Z_0 and Z_1. Add KL divergence between p_0 and p_1 (and $KL(p_1, p_0)$) to loss of g. g will be thrown away after training, as it exists only to increase utility of the flows. The bound holds only when the q estimates are accurate, which is a major limitation.

Fairness with Restricted Encoders: restrict the space of representations to be finite. This allows to get the distribution of sensitive attributes at each z, hence we have $p_i(z)$. First, we bound $P(s=i)$ using binom. conf. intervals, then per-cell balanced accuracy, then BA. This is done on different datasets to achieve independence.

11 Appendix

DM: $\neg(\phi \wedge \psi) = \neg\phi \vee \neg\psi$; $\neg(\phi \vee \psi) = \neg\phi \wedge \neg\psi$

$\mathbb{B}^1_\epsilon \subseteq \mathbb{B}^2_\epsilon \subseteq \mathbb{B}^\infty_\epsilon \subseteq \mathbb{B}^2_{\epsilon\sqrt{d}} \subseteq \mathbb{B}^1_{\epsilon \cdot d}$

Jensen: g convex: $g(E[X]) \leq E[g(X)]$

Bayes: $P(X|Y) = \frac{P(X,Y)}{P(Y)} = \frac{P(Y|X)P(X)}{P(Y)}$

Inv: $A^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$

$||x||_p = \left(\sum_{i=1}^d |x_i|^p\right)^{\frac{1}{p}}$ $||x||_\infty = \max_{i \in \{1,..,d\}} |x_i|$

Softmax $\sigma(z)_i = e^{z_i} / \sum_{j=1}^D e^{z_j}$

CE loss: $CE(\vec{z}, y) = -\sum_{c=1}^K \mathbb{1}[c=y] \cdot \log z_c$

Implication: $\phi \Rightarrow \psi \iff \neg\phi \vee \psi$

Gauss: $\mathcal{N} = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} exp(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu))$

CDF: $\Phi(v; \mu, \sigma^2) = \int_{-\infty}^v \mathcal{N}(y; \mu, \sigma^2) dy = \Phi(\frac{v-\mu}{\sqrt{\sigma^2}}; 0, 1)$

Laplace: $\mathcal{L} = \frac{1}{2b} exp(-\frac{|x-\mu|}{b})$, $\Phi(x; \mu, b) = 0.5 + 0.5\text{sgn}(x-\mu)(1 - \exp(-\frac{|x-\mu|}{b}))$

Subadditivity of $\sqrt{\cdot}$: $\sqrt{x+y} \leq \sqrt{x} + \sqrt{y}$

Cauchy Schwarz: $\langle x, y \rangle \leq ||x||_2 \cdot ||y||_2$

Holdners: $||x \cdot y||_1 \leq ||x||_p \cdot ||y||_q$, if $\frac{1}{p} + \frac{1}{q} = 1$

Minmax: $\max_a \min_b f(a,b) \leq \min_b \max_a f(a,b)$

Variance & Covariance

$\mathbb{V}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$

$\mathbb{V}(X+Y) = \mathbb{V}(X) + \mathbb{V}(Y) + 2\text{Cov}(X,Y)$

$\mathbb{V}(AX) = A\mathbb{V}(X)A^T$, $\mathbb{V}[\alpha X] = \alpha^2 \mathbb{V}[X]$

$\text{Cov}(X,Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$

Distributions

$\text{Exp}(x|\lambda) = \lambda e^{-\lambda x}$, $\text{Ber}(x|\theta) = \theta^x (1-\theta)^{(1-x)}$

Sigmoid: $\sigma(x) = 1/(1+e^{-x})$

$a\mathcal{N}(\mu_1, \sigma_1^2) + \mathcal{N}(\mu_2, \sigma_2^2) = \mathcal{N}(a\mu_1 + \mu_2, a^2\sigma_1^2 + \sigma_2^2)$

Normal CDF

$x \sim \mathcal{N}(0,1) \implies \mathbb{P}(x \leq z) = \Phi(z), \mathbb{P}(x \leq \Phi^{-1}(z)) = z$. $x \sim \mathcal{N}(\mu, \sigma^2) \implies \mathbb{P}(x \leq z) = \Phi(\frac{z-\mu}{\sigma}), \mathbb{P}(x \leq \mu + \sigma\Phi^{-1}(z)) = z$

Chebyshev & Consistency

$\mathbb{P}(|X - \mathbb{E}[X]| \geq \epsilon) \leq \frac{\mathbb{V}[X]}{\epsilon^2}$, $\lim_{n\to\infty} \mathbb{P}(|\hat{\mu} - \mu| > \epsilon) = 0$

Derivatives

$(fg)' = f'g + fg'$; $(f/g)' = (f'g - fg')/g^2$

$f(g(x))' = f'(g(x))g'(x)$; $\log(x)' = 1/x$

$\partial_x \mathbf{b}^\top \mathbf{x} = \partial_x \mathbf{x}^\top \mathbf{b} = \mathbf{b}$, $\partial_x \mathbf{x}^\top \mathbf{x} = \partial_x ||\mathbf{x}||_2^2 = 2\mathbf{x}$,

$\partial_x \mathbf{x}^\top \mathbf{A}\mathbf{x} = (\mathbf{A}^\top + \mathbf{A})\mathbf{x}$, $\partial_x(\mathbf{b}^\top \mathbf{A}\mathbf{x}) = \mathbf{A}^\top \mathbf{b}$,

$\partial_X(\mathbf{c}^\top \mathbf{X}\mathbf{b}) = \mathbf{c}\mathbf{b}^\top$, $\partial_X(\mathbf{c}^\top \mathbf{X}^\top \mathbf{b}) = \mathbf{b}\mathbf{c}^\top$,

$\partial_x(||\mathbf{x} - \mathbf{b}||_2) = \frac{\mathbf{x}-\mathbf{b}}{||\mathbf{x}-\mathbf{b}||_2}$, $\partial_X(||\mathbf{X}||_F^2) = 2\mathbf{X}$,

$\partial_x ||\mathbf{x}||_1 = \frac{\mathbf{x}}{|\mathbf{x}|}$, $\partial_x ||\mathbf{A}\mathbf{x} - \mathbf{b}||_2^2 = 2(\mathbf{A}^\top \mathbf{A}\mathbf{x} - \mathbf{A}^\top \mathbf{b})$,

MILP encodings

$y = |x|, l \leq x \leq u$: $y \geq x, y \geq -x$,
$y \leq -x + a \cdot 2u, y \leq x - (1-a) \cdot 2l$, $a \in \{0,1\}$

$y = \max(x_1, x_2), l_1 \leq x_1 \leq u_1, l_2 \leq x_2 \leq u_2$:
$y \geq x_1, y \geq x_2, y \leq x_1 + a \cdot (u_2 - l_1)$,
$y \leq x_2 + (1-a) \cdot (u_1 - l_2), a \in \{0,1\}$