Are these datasets the same?
Learning kernels for efficient and fair two-sample tests

Danica J. Sutherland (she/her)
University of British Columbia (UBC) / Alberta Machine Intelligence Institute (Amii)

Hsiao-Yu (Fish) Tung
Heiko Strathmann
Soumyajit De
Aaditya Ramdas
Alex Smola
Arthur Gretton

Feng Liu
Wenkai Xu
Jie Lu
Guangquan Zhang
Arthur Gretton

Feng Liu
Wenkai Xu
Jie Lu

Namrata Deka

TrustML - 15 Feb 2022
Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim P$
  - Training error $\approx$ test error on $P$
  - So our model should be good on more samples from $P$
Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - Training error $\approx$ test error on $\mathbb{P}$
  - So our model should be good on more samples from $\mathbb{P}$

- Really:
Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim P$
  - Training error $\approx$ test error on $P$
  - So our model should be good on more samples from $P$

- Really:
  - Train on “i.i.d. samples from some distribution, $X_i \sim P$"
Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - Training error $\approx$ test error on $\mathbb{P}$
  - So our model should be good on more samples from $\mathbb{P}$

- Really:
  - Train on “i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$”
  - Training error might vaguely correlate with test error on $\mathbb{P}$
Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - Training error $\approx$ test error on $\mathbb{P}$
  - So our model should be good on more samples from $\mathbb{P}$

- Really:
  - Train on “i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$”
  - Training error might vaguely correlate with test error on $\mathbb{P}$
  - Deploy it on some distribution $\mathbb{Q}$, might be sort of like $\mathbb{P}$
    - and probably changes over time...
This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

• How is $\mathbb{P}$ different from $\mathbb{Q}$?
This talk

Based on samples \( \{X_i\} \sim P \) and \( \{Y_j\} \sim Q \):

- How is \( P \) different from \( Q \)?
This talk

Based on samples \( \{X_i\} \sim P \) and \( \{Y_j\} \sim Q \):

- How is \( P \) different from \( Q \)?
- Is \( P \) close enough to \( Q \) for our model?
This talk

Based on samples \( \{X_i\} \sim P \) and \( \{Y_j\} \sim Q \):

- How is \( P \) different from \( Q \)?
- Is \( P \) close enough to \( Q \) for our model?
This talk

Based on samples \( \{X_i\} \sim P \) and \( \{Y_j\} \sim Q \):

- How is \( P \) different from \( Q \)?
- Is \( P \) close enough to \( Q \) for our model?
- Is \( P = Q \)?
Two-sample testing

- Given samples from two unknown distributions
  \[ X \sim P \quad Y \sim Q \]

- Question: is \( P = Q \)?
Two-sample testing

- Given samples from two unknown distributions

\[ X \sim P \quad Y \sim Q \]

- Do smokers/non-smokers get different cancers?
Two-sample testing

- Given samples from two unknown distributions

\[ X \sim P \quad Y \sim Q \]

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
Two-sample testing

- Given samples from two unknown distributions $X \sim P$, $Y \sim Q$

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
Two-sample testing

- Given samples from two unknown distributions

\[ X \sim P \quad Y \sim Q \]

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
Two-sample testing

- Given samples from two unknown distributions

\[ X \sim P \quad Y \sim Q \]

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [MMDiff2]
Two-sample testing

- Given samples from two unknown distributions

\[ X \sim P \quad Y \sim Q \]

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? \([\text{MMDiff2}]\)
- Do these dob and birthday columns mean the same thing?
Two-sample testing

- Given samples from two unknown distributions

\[ X \sim P, \quad Y \sim Q \]

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [MMDiff2]
- Do these dob and birthday columns mean the same thing?
- Does my generative model \( Q_\theta \) match \( P_{\text{data}} \)?
Two-sample testing

- Given samples from two unknown distributions

\[ X \sim \mathcal{P} \quad Y \sim \mathcal{Q} \]

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [MMDiff2]
- Do these dob and birthday columns mean the same thing?
- Does my generative model \( Q_\theta \) match \( \mathcal{P}_{\text{data}} \)?
- Independence testing: is \( P(X, Y) = P(X)P(Y) \)?
Two-sample testing

- Given samples from two unknown distributions

\[ X \sim P \quad Y \sim Q \]

- Question: is \( P = Q \)?
Two-sample testing

• Given samples from two unknown distributions

\[ X \sim P \quad Y \sim Q \]

• Question: is \( P = Q \)?

• Hypothesis testing approach:

\[ H_0 : P = Q \quad H_1 : P \neq Q \]
Two-sample testing

- Given samples from two unknown distributions
  \[ X \sim P \quad Y \sim Q \]
- Question: is \( P = Q \)?
- Hypothesis testing approach:
  \[ H_0 : P = Q \quad H_1 : P \neq Q \]
- Reject \( H_0 \) if test statistic \( \hat{T}(X, Y) > c_\alpha \)
What's a hypothesis test again?
What's a hypothesis test again?
don't reject $H_0$  $c_\alpha$  reject $H_0$ (say $P \neq Q$)
What's a hypothesis test again?

don't reject $H_0$  $c_\alpha$  reject $H_0$ (say $P \neq Q$)

false rejection rate: want $\leq \alpha$
What's a hypothesis test again?

don't reject $H_0$  $c_{\alpha}$  reject $H_0$ (say $P \neq Q$)

- $P = Q$
- $P \neq Q$

false rejection rate: want $\leq \alpha$

power: true rejection rate
Permutation testing to find $c_\alpha$

Need $\Pr_{H_0} \left( \hat{T}(X, Y) > c_\alpha \right) \leq \alpha$

$X_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5 \quad Y_1 \quad Y_2 \quad Y_3 \quad Y_4 \quad Y_5$

$c_\alpha: 1 - \alpha$th quantile of $\left\{ \right\}$
Permutation testing to find $c_\alpha$

Need $\Pr_{H_0} \left( \hat{T}(X, Y) > c_\alpha \right) \leq \alpha$

$c_\alpha$: $1 - \alpha$th quantile of $\{ X_1, X_2, X_3, X_4, X_5, Y_1, Y_2, Y_3, Y_4, Y_5 \}$
Permutation testing to find $c_\alpha$

Need $\Pr_{H_0} \left( \hat{T}(X, Y) > c_\alpha \right) \leq \alpha$

$c_\alpha$: $1 - \alpha$th quantile of $\left\{ \hat{T}(\tilde{X}_1, \tilde{Y}_1), \right\}$
Permutation testing to find $c_\alpha$

Need $\Pr_{H_0} \left( \hat{T}(X, Y) > c_\alpha \right) \leq \alpha$

$c_\alpha$: $1 - \alpha$th quantile of $\left\{ \hat{T}(\tilde{X}_1, \tilde{Y}_1), \hat{T}(\tilde{X}_2, \tilde{Y}_2), \right\}$
Permutation testing to find $c_{\alpha}$

Need $\Pr_{H_0} \left( \hat{T}(X, Y) > c_{\alpha} \right) \leq \alpha$

$X_1 \ X_2 \ X_3 \ X_4 \ X_5 \ Y_1 \ Y_2 \ Y_3 \ Y_4 \ Y_5$

$c_{\alpha}: 1 - \alpha$th quantile of $\left\{ \hat{T}(\tilde{X}_1, \tilde{Y}_1), \hat{T}(\tilde{X}_2, \tilde{Y}_2), \cdots \right\}$
Need a $\hat{T}$ to estimate the difference between distributions, based on samples
Need a $\hat{T}$ to estimate the difference between distributions, based on samples

Our choice of $\hat{T}$: the **Maximum Mean Discrepancy (MMD)**
Need a $\hat{T}$ to estimate the difference between distributions, based on samples.

Our choice of $\hat{T}$: the **Maximum Mean Discrepancy (MMD)**

This is a *kernel-based* distance between distributions.
What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \text{sign}(f(x)), f(x) = w^T (x, 1)$
What's a kernel again?

- Linear classifiers: \( \hat{y}(x) = \text{sign}(f(x)), f(x) = w^T (x, 1) \)
What's a kernel again?

- Linear classifiers: \( \hat{y}(x) = \text{sign}(f(x)), f(x) = w^T(x, 1) \)
What's a kernel again?

- Linear classifiers: \( \hat{y}(x) = \text{sign}(f(x)), f(x) = w^T (x, 1) \)
- Use a “richer” \( x \):

\[
f(x) = w^T (x, x^2, 1) = w^T \phi(x)
\]
What's a kernel again?

- Linear classifiers: \( \hat{y}(x) = \text{sign}(f(x)), f(x) = w^T (x, 1) \)
- Use a "richer" \( x \):

\[
f(x) = w^T (x, x^2, 1) = w^T \phi(x)
\]
What's a kernel again?

- Linear classifiers: \( \hat{y}(x) = \text{sign}(f(x)), f(x) = w^\top (x, 1) \)
- Use a “richer” \( x \):
  \[
f(x) = w^\top (x, x^2, 1) = w^\top \phi(x)
\]
What's a kernel again?

- Linear classifiers: \( \hat{y}(x) = \text{sign}(f(x)), f(x) = w^T(x, 1) \)
- Use a “richer” \( x \):
  \[
  f(x) = w^T(x, x^2, 1) = w^T \phi(x)
  \]
- Can avoid explicit \( \phi(x) \); instead \( k(x, y) = \langle \phi(x), \phi(y) \rangle_H \)
What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \text{sign}(f(x)), f(x) = w^T (x, 1)$
- Use a “richer” $x$:
  $$f(x) = w^T (x, x^2, 1) = w^T \phi(x)$$
- Can avoid explicit $\phi(x)$; instead $k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$
- “Kernelized” algorithms access data only through $k(x, y)$
  $$f(x) = \langle w, \phi(x) \rangle_{\mathcal{H}} = \sum_{i=1}^{n} \alpha_i k(X_i, x)$$
What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \text{sign}(f(x))$, $f(x) = w^T (x, 1)$
- Use a “richer” $x$:
  \[ f(x) = w^T (x, x^2, 1) = w^T \phi(x) \]
- Can avoid explicit $\phi(x)$; instead $k(x, y) = \langle \phi(x), \phi(y) \rangle_H$
- “Kernelized” algorithms access data only through $k(x, y)$
  \[ f(x) = \langle w, \phi(x) \rangle_H = \sum_{i=1}^{n} \alpha_i k(X_i, x) \]
- $\|f\|_H = \sqrt{\alpha^T K \alpha}$ gives kernel notion of smoothness
Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF

\[ k(x, y) = \exp \left( -\frac{\|x - y\|^2}{2\sigma^2} \right) \]
Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF

\[ k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right) \]
Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / ...

\[ k(x, y) = \exp \left( -\frac{||x - y||^2}{2\sigma^2} \right) \]
Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / ...

$$k(x, y) = \exp \left( - \frac{\|x - y\|^2}{2\sigma^2} \right)$$

- Some functions with small $\|f\|_\mathcal{H}$:
Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / ...

\[ k(x, y) = \exp \left( -\frac{\|x - y\|^2}{2\sigma^2} \right) \]

- Some functions with small \( \|f\|_\mathcal{H} \):
Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / ...

\[ k(x, y) = \exp \left( - \frac{\|x - y\|^2}{2\sigma^2} \right) \]

- Some functions with small \( \| f \|_H \):
Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / ...

\[ k(x, y) = \exp \left( - \frac{\| x - y \|^2}{2\sigma^2} \right) \]

- Some functions with small \( \| f \|_\mathcal{H} \):
Maximum Mean Discrepancy (MMD)

$$\text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}} [f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}} [f(Y)]$$
Maximum Mean Discrepancy (MMD)

$$\text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}} [f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}} [f(Y)]$$

The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}} [f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}} [f(t)]$
Maximum Mean Discrepancy (MMD)

\[ \text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}}[f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(Y)] \]

The sup is achieved by \( f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)] \)
Maximum Mean Discrepancy (MMD)

$$\text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}} [f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}} [f(Y)]$$

The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}} [f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}} [f(t)]$
Maximum Mean Discrepancy (MMD)

\[
\text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}}[f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(Y)]
\]

The sup is achieved by \( f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)] \)
**Maximum Mean Discrepancy (MMD)**

\[
\text{MMD}_k(P, Q) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim P}[f(X)] - \mathbb{E}_{Y \sim Q}[f(Y)]
\]

The sup is achieved by \( f(t) \propto \mathbb{E}_{X \sim P}[f(t)] - \mathbb{E}_{Y \sim Q}[f(t)] \)
Maximum Mean Discrepancy (MMD)

\[ \text{MMD}_K(P, Q) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim P} [f(X)] - \mathbb{E}_{Y \sim Q} [f(Y)] \]

The sup is achieved by \( f(t) \propto \mathbb{E}_{X \sim P} [f(t)] - \mathbb{E}_{Y \sim Q} [f(t)] \)
Maximum Mean Discrepancy (MMD)

\[ \text{MMD}_k(P, Q) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim P} [f(X)] - \mathbb{E}_{Y \sim Q} [f(Y)] \]

The sup is achieved by \( f(t) \propto \mathbb{E}_{X \sim P} [f(t)] - \mathbb{E}_{Y \sim Q} [f(t)] \)
Maximum Mean Discrepancy (MMD)

\[ \text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}}[f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(Y)] \]

The sup is achieved by \( f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)] \)
Maximum Mean Discrepancy (MMD)

$$\text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}} [f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}} [f(Y)]$$

The sup is achieved by $$f(t) \propto \mathbb{E}_{X \sim \mathbb{P}} [f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}} [f(t)]$$
Maximum Mean Discrepancy (MMD)

\[ \text{MMD}_k(P, Q) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim P}[f(X)] - \mathbb{E}_{Y \sim Q}[f(Y)] \]

The sup is achieved by \( f(t) \propto \mathbb{E}_{X \sim P}[f(t)] - \mathbb{E}_{Y \sim Q}[f(t)] \)
Maximum Mean Discrepancy (MMD)

\[
\operatorname{MMD}_k(P, Q) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim P} [f(X)] - \mathbb{E}_{Y \sim Q} [f(Y)]
\]

The sup is achieved by \( f(t) \propto \mathbb{E}_{X \sim P} [f(t)] - \mathbb{E}_{Y \sim Q} [f(t)] \)

\[
\operatorname{MMD}^2(P, Q) = \mathbb{E}_{X, X' \sim P} [k(X, X') + k(Y, Y') - 2k(X, Y)]
\]
\[ \text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}}[f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(Y)] \]
$$\text{MMD}_k(\mathcal{P}, \mathcal{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathcal{P}} [f(X)] - \mathbb{E}_{Y \sim \mathcal{Q}} [f(Y)]$$

$$= \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathcal{P}} [\langle f, \varphi(X) \rangle_{\mathcal{H}}] - \mathbb{E}_{Y \sim \mathcal{Q}} [\langle f, \varphi(Y) \rangle_{\mathcal{H}}]$$
\[
\text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}} [f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}} [f(Y)]
\]

\[
= \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}} [\langle f, \varphi(X) \rangle_{\mathcal{H}}] - \mathbb{E}_{Y \sim \mathbb{Q}} [\langle f, \varphi(Y) \rangle_{\mathcal{H}}]
\]

\[
= \sup_{\|f\|_{\mathcal{H}} \leq 1} \left\langle f, \mathbb{E}_{X \sim \mathbb{P}} [\varphi(X)] - \mathbb{E}_{Y \sim \mathbb{Q}} [\varphi(Y)] \right\rangle_{\mathcal{H}}
\]
\[ \text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_\mathcal{H} \leq 1} \mathbb{E}[f(X)] - \mathbb{E}[f(Y)] \]

\[ = \sup_{\|f\|_\mathcal{H} \leq 1} \mathbb{E} \left[ \langle f, \varphi(X) \rangle_\mathcal{H} \right] - \mathbb{E} \left[ \langle f, \varphi(Y) \rangle_\mathcal{H} \right] \]

\[ = \sup_{\|f\|_\mathcal{H} \leq 1} \left\langle f, \mathbb{E}_X[\varphi(X)] - \mathbb{E}_Y[\varphi(Y)] \right\rangle_\mathcal{H} \]

\[ = \sup_{\|f\|_\mathcal{H} \leq 1} \left\langle f, \mu^k_\mathbb{P} - \mu^k_\mathbb{Q} \right\rangle_\mathcal{H} \]
\[
\text{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}} [f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}} [f(Y)] \\
= \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}} [\langle f, \varphi(X) \rangle_{\mathcal{H}}] - \mathbb{E}_{Y \sim \mathbb{Q}} [\langle f, \varphi(Y) \rangle_{\mathcal{H}}] \\
= \sup_{\|f\|_{\mathcal{H}} \leq 1} \left\langle f, \mathbb{E}_{X \sim \mathbb{P}} [\varphi(X)] - \mathbb{E}_{Y \sim \mathbb{Q}} [\varphi(Y)] \right\rangle_{\mathcal{H}} \\
= \sup_{\|f\|_{\mathcal{H}} \leq 1} \left\langle f, \mu^k_{\mathbb{P}} - \mu^k_{\mathbb{Q}} \right\rangle_{\mathcal{H}} = \|\mu^k_{\mathbb{P}} - \mu^k_{\mathbb{Q}}\|_{\mathcal{H}}
\]
\[ \text{MMD}_k(P, Q) = \sup_{\|f\|_\mathcal{H} \leq 1} \mathbb{E}_{X \sim P} [f(X)] - \mathbb{E}_{Y \sim Q} [f(Y)] \]

\[ = \sup_{\|f\|_\mathcal{H} \leq 1} \mathbb{E}_{X \sim P} [\langle f, \varphi(X) \rangle_\mathcal{H}] - \mathbb{E}_{Y \sim Q} [\langle f, \varphi(Y) \rangle_\mathcal{H}] \]

\[ = \sup_{\|f\|_\mathcal{H} \leq 1} \left\langle f, \mathbb{E}_{X \sim P} [\varphi(X)] - \mathbb{E}_{Y \sim Q} [\varphi(Y)] \right\rangle_\mathcal{H} \]

\[ = \sup_{\|f\|_\mathcal{H} \leq 1} \left\langle f, \mu_P^k - \mu_Q^k \right\rangle_\mathcal{H} = \|\mu_P^k - \mu_Q^k\|_\mathcal{H} \]

\[ \langle \mu_P^k, \mu_Q^k \rangle_\mathcal{H} = \mathbb{E}_{X \sim P} \langle \varphi(X), \varphi(Y) \rangle_\mathcal{H} = \mathbb{E}_{X \sim P} k(X, Y) \]
\[
\text{MMD}_k(P, Q) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim P} [f(X)] - \mathbb{E}_{Y \sim Q} [f(Y)] \\
= \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim P} [\langle f, \varphi(X) \rangle_{\mathcal{H}}] - \mathbb{E}_{Y \sim Q} [\langle f, \varphi(Y) \rangle_{\mathcal{H}}] \\
= \sup_{\|f\|_{\mathcal{H}} \leq 1} \left\langle f, \mathbb{E}_{X \sim P} [\varphi(X)] - \mathbb{E}_{Y \sim Q} [\varphi(Y)] \right\rangle_{\mathcal{H}} \\
= \sup_{\|f\|_{\mathcal{H}} \leq 1} \left\langle f, \mu_P^k - \mu_Q^k \right\rangle_{\mathcal{H}} = \|\mu_P^k - \mu_Q^k\|_{\mathcal{H}} \\
\langle \mu_P^k, \mu_Q^k \rangle_{\mathcal{H}} = \mathbb{E}_{X \sim P, Y \sim Q} \langle \varphi(X), \varphi(Y) \rangle_{\mathcal{H}} = \mathbb{E}_{X \sim P, Y \sim Q} k(X, Y) \\
\text{MMD}^2(P, Q) = \mathbb{E}_{X, X' \sim P, Y, Y' \sim Q} [k(X, X') + k(Y, Y') - 2k(X, Y)]
\]
Estimating MMD

$$\text{MMD}_k^2(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{X, X' \sim \mathbb{P}}[k(X, X')] + \mathbb{E}_{Y, Y' \sim \mathbb{Q}}[k(Y, Y')] - 2 \mathbb{E}_{X \sim \mathbb{P}}[k(X, Y)]$$
Estimating MMD

\[
\text{MMD}_k^2(\mathcal{P}, \mathcal{Q}) = \mathbb{E}_{X,X' \sim \mathcal{P}}[k(X, X')] + \mathbb{E}_{Y,Y' \sim \mathcal{Q}}[k(Y, Y')] - 2 \mathbb{E}_{X \sim \mathcal{P}, Y \sim \mathcal{Q}}[k(X, Y)]
\]

\[
\widehat{\text{MMD}}_k^2(X, Y) = \text{mean}(K_{XX}) + \text{mean}(K_{YY}) - 2 \text{mean}(K_{XY})
\]
Estimating MMD

\[
\text{MMD}_k^2(\mathcal{P}, \mathcal{Q}) = \mathbb{E}_{X,X' \sim \mathcal{P}} [k(X, X')] + \mathbb{E}_{Y,Y' \sim \mathcal{Q}} [k(Y, Y')] - 2 \mathbb{E}_{X \sim \mathcal{P}} \mathbb{E}_{Y \sim \mathcal{Q}} [k(X, Y)]
\]

\[
\widehat{\text{MMD}}_k^2(\mathbf{X}, \mathbf{Y}) = \text{mean}(K_{XX}) + \text{mean}(K_{YY}) - 2 \text{mean}(K_{XY})
\]
Estimating MMD

\[
\text{MMD}_k^2(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{X,X' \sim \mathbb{P}} [k(X, X')] + \mathbb{E}_{Y,Y' \sim \mathbb{Q}} [k(Y, Y')] - 2 \mathbb{E}_{X \sim \mathbb{P}, Y \sim \mathbb{Q}} [k(X, Y)]
\]

\[
\widehat{\text{MMD}}_k^2(X, Y) = \text{mean}(K_{XX}) + \text{mean}(K_{YY}) - 2 \text{mean}(K_{XY})
\]
Estimating MMD

\[
\text{MMD}_k^2(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{X,X' \sim \mathbb{P}}[k(X, X')] + \mathbb{E}_{Y,Y' \sim \mathbb{Q}}[k(Y, Y')] - 2\mathbb{E}_{X \sim \mathbb{P}, Y \sim \mathbb{Q}}[k(X, Y)]
\]

\[
\widehat{\text{MMD}}_k^2(X, Y) = \text{mean}(K_{XX}) + \text{mean}(K_{YY}) - 2\text{mean}(K_{XY})
\]
MMD as feature matching

$$MMD_k(P, Q) = \left\| \mathbb{E}_{X \sim P} [\varphi(X)] - \mathbb{E}_{Y \sim Q} [\varphi(Y)] \right\|_{\mathcal{H}}$$

- $\varphi : X \rightarrow \mathcal{H}$ is the feature map for $k(x, y) = \langle \varphi(x), \varphi(y) \rangle$
MMD as feature matching

\[
\text{MMD}_k(\mathcal{P}, \mathcal{Q}) = \left\| \mathbb{E}_{X \sim \mathcal{P}}[\varphi(X)] - \mathbb{E}_{Y \sim \mathcal{Q}}[\varphi(Y)] \right\|_H
\]

- \( \varphi : X \rightarrow \mathcal{H} \) is the feature map for \( k(x, y) = \langle \varphi(x), \varphi(y) \rangle \)
- If \( k(x, y) = x^T y \), \( \varphi(x) = x \), then the MMD is distance between means
MMD as feature matching

\[ \text{MMD}_k(P, Q) = \left\| \mathbb{E}_{X \sim P} [\varphi(X)] - \mathbb{E}_{Y \sim Q} [\varphi(Y)] \right\|_{\mathcal{H}} \]

- \( \varphi : X \rightarrow \mathcal{H} \) is the feature map for \( k(x, y) = \langle \varphi(x), \varphi(y) \rangle \)
- If \( k(x, y) = x^T y \), \( \varphi(x) = x \), then the MMD is distance between means
- Many kernels: \text{infinite-dimensional} \ \mathcal{H}
MMD-based tests

- If $k$ is characteristic, $\text{MMD}(\mathcal{P}, \mathcal{Q}) = 0$ iff $\mathcal{P} = \mathcal{Q}$

- Efficient permutation testing for $\widehat{\text{MMD}}(\mathbf{X}, \mathbf{Y})$
MMD-based tests

- If $k$ is characteristic, $\text{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$

- Efficient permutation testing for $\widehat{\text{MMD}}(X, Y)$
  - $H_0$: $n\widehat{\text{MMD}}^2$ converges in distribution
  - $H_1$: $\sqrt{n}(\widehat{\text{MMD}}^2 - \text{MMD}^2)$ asymptotically normal
MMD-based tests

- If $k$ is characteristic, $\text{MMD}(P, Q) = 0$ iff $P = Q$

- Efficient permutation testing for $\widehat{\text{MMD}}(X, Y)$
  - $H_0$: $n\widehat{\text{MMD}}^2$ converges in distribution
  - $H_1$: $\sqrt{n}(\widehat{\text{MMD}}^2 - \text{MMD}^2)$ asymptotically normal

- Any characteristic kernel gives consistent test
MMD-based tests

- If $k$ is characteristic, $\text{MMD}(\mathcal{P}, \mathcal{Q}) = 0$ iff $\mathcal{P} = \mathcal{Q}$

- Efficient permutation testing for $\widehat{\text{MMD}}(X, Y)$
  - $H_0$: $n\hat{\text{MMD}}^2$ converges in distribution
  - $H_1$: $\sqrt{n}(\text{MMD}^2 - \text{MMD}^2)$ asymptotically normal

- Any characteristic kernel gives consistent test...eventually
MMD-based tests

- If $k$ is characteristic, $\text{MMD}(P, Q) = 0$ iff $P = Q$

- Efficient permutation testing for $\hat{\text{MMD}}(X, Y)$
  - $H_0$: $n\hat{\text{MMD}}^2$ converges in distribution
  - $H_1$: $\sqrt{n}(\hat{\text{MMD}}^2 - \text{MMD}^2)$ asymptotically normal

- Any characteristic kernel gives consistent test...eventually

- Need enormous $n$ if kernel is bad for problem
Classifier two-sample tests

- $\hat{T}(X, Y)$ is the accuracy of $f$ on the test set
- Under $H_0$, classification impossible: $\hat{T} \sim \text{Binomial}(n, \frac{1}{2})$
Classifier two-sample tests

- $\hat{T}(X, Y)$ is the accuracy of $f$ on the test set
- Under $H_0$, classification impossible: $\hat{T} \sim \text{Binomial}(n, \frac{1}{2})$
- With $k(x, y) = \frac{1}{4} f(x) f(y)$ where $f(x) \in \{-1, 1\}$, get $	ext{MMD}(X, Y) = |\hat{T}(X, Y) - \frac{1}{2}|$
Deep learning and deep kernels

- $k(x, y) = \frac{1}{4} f(x) f(y)$ is one form of deep kernel
Deep learning and deep kernels

- $k(x, y) = \frac{1}{4} f(x) f(y)$ is one form of deep kernel.

- Deep models are usually of the form $f(x) = w^T \phi_\psi(x)$
  - With a learned $\phi_\psi(x) : \mathcal{X} \rightarrow \mathbb{R}^D$
Deep learning and deep kernels

- $k(x, y) = \frac{1}{4} f(x) f(y)$ is one form of deep kernel

- Deep models are usually of the form $f(x) = w^T \phi_\psi(x)$
  - With a learned $\phi_\psi(x) : \mathcal{X} \rightarrow \mathbb{R}^D$

- If we fix $\psi$, have $f \in \mathcal{H}_\psi$ with $k_\psi(x, y) = \phi_\psi(x)^T \phi_\psi(y)$
Deep learning and deep kernels

• $k(x, y) = \frac{1}{4} f(x) f(y)$ is one form of deep kernel

• Deep models are usually of the form $f(x) = w^T \phi_\psi(x)$
  ▪ With a learned $\phi_\psi(x) : X \rightarrow \mathbb{R}^D$

• If we fix $\psi$, have $f \in \mathcal{H}_\psi$ with $k_\psi(x, y) = \phi_\psi(x)^T \phi_\psi(y)$
  ▪ Same idea as NNGP approximation
Deep learning and deep kernels

- $k(x, y) = \frac{1}{4} f(x) f(y)$ is one form of deep kernel

- Deep models are usually of the form $f(x) = w^T \phi_\psi(x)$
  - With a learned $\phi_\psi(x) : \mathcal{X} \rightarrow \mathbb{R}^D$

- If we fix $\psi$, have $f \in \mathcal{H}_\psi$ with $k_\psi(x, y) = \phi_\psi(x)^T \phi_\psi(y)$
  - Same idea as NNGP approximation

- Generalize to a deep kernel:
  $$k_\psi(x, y) = \kappa(\phi_\psi(x), \phi_\psi(y))$$
Normal deep learning ⊂ deep kernels

• Take $k_\psi(x, y) = \frac{1}{4} f_\psi(x) f_\psi(y)$

• Final function in $\mathcal{H}_\psi$ will be $a f_\psi(x)$
Normal deep learning $\subset$ deep kernels

- Take $k_\psi(x, y) = \frac{1}{4} f_\psi(x) f_\psi(y) + 1$
- Final function in $\mathcal{H}_\psi$ will be $a f_\psi(x) + b$
Normal deep learning ⊆ deep kernels

- Take $k_\psi(x, y) = \frac{1}{4} f_\psi(x)f_\psi(y) + 1$
- Final function in $\mathcal{H}_\psi$ will be $a f_\psi(x) + b$
- With logistic loss: this is Platt scaling
Normal deep learning $\subset$ deep kernels

- Take $k_\psi(x, y) = \frac{1}{4} f_\psi(x) f_\psi(y) + 1$
- Final function in $\mathcal{H}_\psi$ will be $a f_\psi(x) + b$
- With logistic loss: this is Platt scaling

On Calibration of Modern Neural Networks

Chuan Guo*1  Geoff Pleiss*1  Yu Sun*1  Kilian Q. Weinberger1
So what?

- This definitely does *not* say that deep learning is (even approximately) a kernel method
So what?

- This definitely does *not* say that deep learning is (even approximately) a kernel method
- ...despite what some people might want you to think

- **Computer Science > Machine Learning**
  
  *Submitted on 30 Nov 2020*
  
  **Every Model Learned by Gradient Descent Is Approximately a Kernel Machine**
  
  **Pedro Domingos**
So what?

- This definitely does not say that deep learning is (even approximately) a kernel method
- ...despite what some people might want you to think

**Computer Science > Machine Learning**

*Submitted on 30 Nov 2020*

**Every Model Learned by Gradient Descent Is Approximately a Kernel Machine**

Pedro Domingos

- We know theoretically deep learning can learn some things faster than any kernel method [see Malach+ ICML-21 + refs]
So what?

• This definitely does not say that deep learning is (even approximately) a kernel method

• ...despite what some people might want you to think

We know theoretically deep learning can learn some things faster than any kernel method [see Malach+ ICML-21 + refs]

• But deep kernel learning ≠ traditional kernel models
  ■ exactly like how usual deep learning ≠ linear models
Optimizing power of MMD tests

- Asymptotics of $\hat{\text{MMD}}^2$ give us immediately that

$$\Pr_{H_1} \left( n\hat{\text{MMD}}^2 > c_\alpha \right) \approx \Phi \left( \frac{\sqrt{n} \text{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}} \right)$$

$\text{MMD}$, $\sigma_{H_1}$, $c_\alpha$ are constants: first term usually dominates
Optimizing power of MMD tests

- Asymptotics of $\hat{\text{MMD}}^2$ give us immediately that

$$\Pr_{H_1} \left( n \hat{\text{MMD}}^2 > c_\alpha \right) \approx \Phi \left( \frac{\sqrt{n} \text{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n} \sigma_{H_1}} \right)$$

$\text{MMD}, \sigma_{H_1}, c_\alpha$ are constants: first term usually dominates

- Pick $k$ to maximize an estimate of $\text{MMD}^2 / \sigma_{H_1}$
Optimizing power of MMD tests

- Asymptotics of \( \hat{\text{MMD}}^2 \) give us immediately that

\[
\Pr_{H_1} \left( n\hat{\text{MMD}}^2 > c_\alpha \right) \approx \Phi \left( \frac{\sqrt{n} \text{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}} \right)
\]

\( \text{MMD}, \sigma_{H_1}, c_\alpha \) are constants: first term usually dominates

- Pick \( k \) to maximize an estimate of \( \text{MMD}^2 / \sigma_{H_1} \)

- Use \( \hat{\text{MMD}} \) from before, get \( \hat{\sigma}_{H_1} \) from U-statistic theory
Optimizing power of MMD tests

- Asymptotics of $\widehat{\text{MMD}}^2$ give us immediately that

$$\Pr_{H_1} \left( n\widehat{\text{MMD}}^2 > c_\alpha \right) \approx \Phi \left( \frac{\sqrt{n} \text{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}} \right)$$

$\text{MMD}, \sigma_{H_1}, c_\alpha$ are constants: first term usually dominates

- Pick $k$ to maximize an estimate of $\text{MMD}^2 / \sigma_{H_1}$

- Use $\widehat{\text{MMD}}$ from before, get $\hat{\sigma}_{H_1}$ from U-statistic theory

- Can show uniform $O_P \left( n^{-\frac{1}{3}} \right)$ convergence of estimator
Blobs dataset
Blobs kernels
Investigating a GAN on MNIST

$MMD^2 = 0.0001$
CIFAR-10 vs CIFAR-10.1

Train on 1 000, test on 1 031, repeat 10 times. Rejection rates:

<table>
<thead>
<tr>
<th></th>
<th>ME</th>
<th>SCF</th>
<th>C2ST</th>
<th>MMD-O</th>
<th>MMD-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rejection rate</td>
<td>0.588</td>
<td>0.171</td>
<td>0.452</td>
<td>0.316</td>
<td>0.744</td>
</tr>
</tbody>
</table>
# Ablation vs classifier-based tests

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Cross-entropy</th>
<th></th>
<th></th>
<th>Max power</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Sign</td>
<td>Lin</td>
<td>Ours</td>
<td>Sign</td>
<td>Lin</td>
<td>Ours</td>
</tr>
<tr>
<td>Bloomberg</td>
<td>0.84</td>
<td>0.94</td>
<td>0.90</td>
<td>—</td>
<td>0.95</td>
<td>0.99</td>
</tr>
<tr>
<td>High-d Gauss. mix.</td>
<td>0.47</td>
<td>0.59</td>
<td>0.29</td>
<td>—</td>
<td>0.64</td>
<td>0.66</td>
</tr>
<tr>
<td>Higgs</td>
<td>0.26</td>
<td>0.40</td>
<td>0.35</td>
<td>—</td>
<td>0.30</td>
<td>0.40</td>
</tr>
<tr>
<td>MNIST vs GAN</td>
<td>0.65</td>
<td>0.71</td>
<td>0.80</td>
<td>—</td>
<td>0.94</td>
<td>1.00</td>
</tr>
</tbody>
</table>
But...

• What if you don't have much data for your testing problem?
But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
- Also need enough test data to actually detect the difference
But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
- Also need enough test data to actually detect the difference
- Best split depends on best kernel's quality / how hard to find
But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
- Also need enough test data to actually detect the difference
- Best split depends on best kernel's quality / how hard to find
  - Don't know that ahead of time; can't try more than one
One idea: what if we have *related* problems?
Meta-testing

- One idea: what if we have related problems?
- Similar setup to meta-learning:

\[ B \]

(from Wei+ 2018)
Meta-testing for CIFAR-10 vs CIFAR-10.1

- CIFAR-10 has 60,000 images, but CIFAR-10.1 only has 2,031.
- Where do we get related data from?
Meta-testing for CIFAR-10 vs CIFAR-10.1

- CIFAR-10 has 60,000 images, but CIFAR-10.1 only has 2,031
- Where do we get related data from?
- One option: set up tasks to distinguish classes of CIFAR-10 (airplane vs automobile, airplane vs bird, ...)

One approach (MAML-like)

$A_\theta$ is, e.g., 5 steps of gradient descent we learn the initialization, maybe step size, etc.

$\arg\max_{A_\theta} J(\text{samples}; A_\theta(\text{samples}))$
One approach (MAML-like)

\( A_\theta \) is, e.g., 5 steps of gradient descent
we learn the initialization, maybe step size, etc

This works, but not as well as we'd hoped...
Initialization might work okay on everything, not really adapt
Another approach: Meta-MKL

Inspired by classic multiple kernel learning

Only need to learn linear combination $\beta_i$ on test task: much easier
Theoretical analysis for Meta-MKL

- Same big-O dependence on test task size 😞
- But multiplier is *much* better:
  based on number of meta-training tasks, not on network size
Theoretical analysis for Meta-MKL

- Same big-O dependence on test task size 😞
- But multiplier is *much* better: based on number of meta-training tasks, not on network size
- Coarse analysis: assumes one meta-tasks is “related” enough
  - We compete with picking the single best related kernel
  - Haven't analyzed meaningfully combining related kernels (yet!)
Results on CIFAR-10.1

<table>
<thead>
<tr>
<th>Methods</th>
<th>$m_{tr} = 100$</th>
<th></th>
<th>$m_{tr} = 200$</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$m_{te} = 200$</td>
<td>$m_{te} = 500$</td>
<td>$m_{te} = 900$</td>
<td>$m_{te} = 200$</td>
</tr>
<tr>
<td>ME</td>
<td>0.084±0.009</td>
<td>0.096±0.016</td>
<td>0.160±0.035</td>
<td>0.104±0.013</td>
</tr>
<tr>
<td>SCF</td>
<td>0.047±0.013</td>
<td>0.037±0.011</td>
<td>0.047±0.015</td>
<td>0.026±0.009</td>
</tr>
<tr>
<td>C2ST-S</td>
<td>0.059±0.009</td>
<td>0.062±0.007</td>
<td>0.059±0.007</td>
<td>0.052±0.011</td>
</tr>
<tr>
<td>C2ST-L</td>
<td>0.064±0.009</td>
<td>0.064±0.006</td>
<td>0.063±0.007</td>
<td>0.075±0.014</td>
</tr>
<tr>
<td>MMD-O</td>
<td>0.091±0.011</td>
<td>0.141±0.009</td>
<td>0.279±0.018</td>
<td>0.084±0.007</td>
</tr>
<tr>
<td>MMD-D</td>
<td>0.104±0.007</td>
<td>0.222±0.020</td>
<td>0.418±0.046</td>
<td>0.117±0.013</td>
</tr>
<tr>
<td>AGT-KL</td>
<td>0.170±0.032</td>
<td>0.457±0.052</td>
<td>0.765±0.045</td>
<td>0.152±0.023</td>
</tr>
<tr>
<td>Meta-KL</td>
<td>0.245±0.010</td>
<td>0.671±0.026</td>
<td>0.959±0.013</td>
<td>0.226±0.015</td>
</tr>
<tr>
<td>Meta-MKL</td>
<td><strong>0.277±0.016</strong></td>
<td><strong>0.728±0.020</strong></td>
<td><strong>0.973±0.008</strong></td>
<td><strong>0.255±0.020</strong></td>
</tr>
</tbody>
</table>
But...

- Sometimes we know ahead of time that there are differences that we don't care about
But...

- Sometimes we know ahead of time that there are differences that we don't care about
  - In the MNIST GAN criticism, initial attempt just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has
But...

- Sometimes we know ahead of time that there are differences that we don't care about
  - In the MNIST GAN criticism, initial attempt just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has

- Can we find a kernel that *can* distinguish $P^t$ from $Q^t$, but *can't* distinguish $P^s$ from $Q^s$?
But...

- Sometimes we know ahead of time that there are differences that we don't care about
  - In the MNIST GAN criticism, initial attempt just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has

- Can we find a kernel that *can* distinguish $P^t$ from $Q^t$, but *can't* distinguish $P^s$ from $Q^s$?

- Also useful for *fair representation learning*
But...

- Sometimes we know ahead of time that there are differences that we don't care about
  - In the MNIST GAN criticism, initial attempt just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has

- Can we find a kernel that *can* distinguish $P^t$ from $Q^t$, but *can't* distinguish $P^s$ from $Q^s$?

- Also useful for **fair representation learning**
  - e.g. can distinguish “creditworthy” vs not, can't distinguish by race
High on one power, low on another

Choose $k$ with $\min_k \rho_k^s - \rho_k^t$
High on one power, low on another

Choose $k$ with $\min_k \rho^s_k - \rho^t_k$

- First idea: $\rho = \frac{(\text{MMD})^2}{\sigma_{H_1}}$
High on one power, low on another

\[ \text{Choose } k \text{ with } \min_k \rho_k^s - \rho_k^t \]

- First idea: \[ \rho = \frac{(\text{MMD})^2}{\sigma_{H_1}} \]
  - No good: doesn't balance power appropriately
High on one power, low on another

Choose $k$ with $\min_k \rho_k^s - \rho_k^t$

• First idea: $\rho = \frac{(\text{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately

• Second idea: $\rho = \Phi \left( \frac{\sqrt{n}(\text{MMD})^2 - c_\alpha}{\sigma_{H_1}} \right)$
High on one power, low on another

Choose \( k \) with \( \min_k \rho_k^s - \rho_k^t \)

- First idea: \( \rho = \frac{(\text{MMD})^2}{\sigma_{H_1}} \)
  - No good: doesn't balance power appropriately

- Second idea: \( \rho = \Phi \left( \frac{\sqrt{n}(\text{MMD})^2 - c_\alpha}{\sigma_{H_1}} \right) \)
  - Can estimate \( c_\alpha \) inside the optimization
High on one power, low on another

Choose $k$ with $\min_k \rho^s_k - \rho^t_k$

• First idea: $\rho = \frac{(\text{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately

• Second idea: $\rho = \Phi \left( \frac{\sqrt{n}(\text{MMD})^2 - c_\alpha}{\sigma_{H_1}} \right)$
  - Can estimate $c_\alpha$ inside the optimization
  - Better, but tends to “stall out” in minimizing $\rho^s_k$
Block estimator [Zaremba+ NeurIPS-13]

- Use previous $\widehat{\text{MMD}}$ on $b$ blocks, each of size $B$

- Final estimator: average of each block's estimate
Block estimator [Zaremba+ NeurIPS-13]

- Use previous $\widehat{\text{MMD}}$ on $b$ blocks, each of size $B$

- Final estimator: average of each block's estimate
  - Each block has previous asymptotics
Block estimator \([\text{Zaremba+ NeurIPS-13}]\)

- Use previous \(\hat{\text{MMD}}\) on \(b\) blocks, each of size \(B\)

- Final estimator: average of each block's estimate
  - Each block has previous asymptotics
  - Central limit theorem across blocks
Block estimator [Zaremba+ NeurIPS-13]

- Use previous \( \widehat{\text{MMD}} \) on \( b \) blocks, each of size \( B \)

- Final estimator: average of each block's estimate
  - Each block has previous asymptotics
  - Central limit theorem across blocks

- Power is \( \rho = \Phi \left( \sqrt{bB} \frac{\text{MMD}^2}{\sigma^2_{H_1}} - \Phi^{-1}(1 - \alpha) \right) \)
MMD-B-Fair

• Choose $k$ as $\min_k \rho_k^s - \rho_k^t$
MMD-B-Fair

- Choose $k$ as $\min_k \rho_k^s - \rho_k^t$
  - $\rho$ is the power of a test with $b$ blocks of size $B$
MMD-B-Fair

• Choose $k$ as $\min_k \rho_k^s - \rho_k^t$
  ▪ $\rho$ is the power of a test with $b$ blocks of size $B$
  ▪ We don't actually use a block estimator computationally
MMD-B-Fair

- Choose $k$ as $\min_k \rho^s_k - \rho^t_k$
  - $\rho$ is the power of a test with $b$ blocks of size $B$
  - We don't actually use a block estimator computationally
  - $b, B$ have nothing to do with minibatch size
MMD-B-Fair

- Choose $k$ as $\min_k \rho_k^s - \rho_k^t$
  - $\rho$ is the power of a test with $b$ blocks of size $B$
  - We *don’t* actually use a block estimator computationally
  - $b, B$ have *nothing to do* with minibatch size
- Representation learning: $\min_{\phi} \max_{\kappa} \rho_{\kappa \circ \phi}^s - \rho_{\kappa \circ \phi}^t$
**MMD-B-Fair**

- Choose $k$ as $\min_k \rho^s_k - \rho^t_k$
  - $\rho$ is the power of a test with $b$ blocks of size $B$
  - We *don’t* actually use a block estimator computationally
  - $b$, $B$ have *nothing to do* with minibatch size
- Representation learning: $\min_\phi \max_\kappa \rho^s_{\kappa \circ \phi} - \rho^t_{\kappa \circ \phi}$
  - Deep kernel is $[\kappa \circ \phi](x, y) = \kappa(\phi(x), \phi(y))$
MMD-B-Fair

- Choose $k$ as $\min_k \rho^s_k - \rho^t_k$
  - $\rho$ is the power of a test with $b$ blocks of size $B$
  - We don't actually use a block estimator computationally
  - $b$, $B$ have nothing to do with minibatch size

- Representation learning: $\min_{\phi} \max_{\kappa} \rho^s_{\kappa \circ \phi} - \rho^t_{\kappa \circ \phi}$
  - Deep kernel is $[\kappa \circ \phi](x, y) = \kappa(\phi(x), \phi(y))$
  - $\kappa$ could be deep itself, with adversarial optimization
MMD-B-Fair

- Choose $k$ as $\min_k \rho^s_k - \rho^t_k$
  - $\rho$ is the power of a test with $b$ blocks of size $B$
  - We don't actually use a block estimator computationally
  - $b, B$ have nothing to do with minibatch size

- Representation learning: $\min_\phi \max_\kappa \rho^s_{\kappa \circ \phi} - \rho^t_{\kappa \circ \phi}$
  - Deep kernel is $[\kappa \circ \phi](x, y) = \kappa(\phi(x), \phi(y))$
  - $\kappa$ could be deep itself, with adversarial optimization
  - For now, just Gaussians with different lengthscales
## Adult Data Set

*Download: [Data Folder](#), [Data Set Description](#)*

**Abstract:** Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset.

<table>
<thead>
<tr>
<th>Data Set Characteristics:</th>
<th>Multivariate</th>
<th>Number of Instances:</th>
<th>48842</th>
<th>Area:</th>
<th>Social</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Characteristics:</td>
<td>Categorical, Integer</td>
<td>Number of Attributes:</td>
<td>14</td>
<td>Date Donated</td>
<td>1996-05-01</td>
</tr>
<tr>
<td>Associated Tasks:</td>
<td>Classification</td>
<td>Missing Values?:</td>
<td>Yes</td>
<td>Number of Web Hits:</td>
<td>2390574</td>
</tr>
</tbody>
</table>
**Adult Data Set**

*Download:* Data Folder, Data Set Description

Abstract: Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset.

<table>
<thead>
<tr>
<th>Data Set Characteristics:</th>
<th>Multivariate</th>
<th>Number of Instances:</th>
<th>48842</th>
<th>Area:</th>
<th>Social</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attribute Characteristics:</td>
<td>Categorical, Integer</td>
<td>Number of Attributes:</td>
<td>14</td>
<td>Date Donated</td>
<td>1996-05-01</td>
</tr>
<tr>
<td>Associated Tasks:</td>
<td>Classification</td>
<td>Missing Values?:</td>
<td>Yes</td>
<td>Number of Web Hits:</td>
<td>2390574</td>
</tr>
</tbody>
</table>

---

**Shapes3D**

\[ P^t: \]
\[ Q^t: \]

\[ P^s: \]
\[ Q^s: \]
<table>
<thead>
<tr>
<th>ci-ratio</th>
<th>Method</th>
<th>$\text{Pr(\text{target})}$</th>
<th>$\text{Pr(\text{sensitive})}$</th>
<th>$\text{Pr(\text{sensitive})}$ fine-tuned</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.1, 0.1)</td>
<td>Laftr</td>
<td>0.2500</td>
<td>0.6100</td>
<td>1.000 ($\sigma = 0.111$)</td>
</tr>
<tr>
<td></td>
<td>Cfair</td>
<td>0.2500</td>
<td>0.6071</td>
<td><strong>0.8929 ($\sigma = 0.087$)</strong></td>
</tr>
<tr>
<td></td>
<td>Ffvae</td>
<td>0.1785</td>
<td>0.6428</td>
<td>1.000 ($\sigma = 0.0695$)</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td><strong>1.000</strong></td>
<td><strong>0.2500</strong></td>
<td>0.9642 ($\sigma = 0.007$)</td>
</tr>
<tr>
<td>(0.33, 0.66)</td>
<td>Laftr</td>
<td>0.285</td>
<td>0.607</td>
<td>1.000 ($\sigma = 0.237$)</td>
</tr>
<tr>
<td></td>
<td>Cfair</td>
<td>0.2857</td>
<td>0.6071</td>
<td>1.000 ($\sigma = 0.234$)</td>
</tr>
<tr>
<td></td>
<td>Ffvae</td>
<td>0.9642</td>
<td>1.000</td>
<td>1.000 ($\sigma = 0.075$)</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>1.000</td>
<td><strong>0.5614</strong></td>
<td><strong>0.6842 ($\sigma = 0.005$)</strong></td>
</tr>
</tbody>
</table>

(a) **Adult dataset:** Our method outperforms all others even when additional layers are trained to maximize the sensitive power (albeit with smaller bandwidths in the under-represented scenario).

<table>
<thead>
<tr>
<th>ci-ratio</th>
<th>Method</th>
<th>$\text{Pr(\text{target})}$</th>
<th>$\text{Pr(\text{sensitive})}$</th>
<th>$\text{Pr(\text{sensitive})}$ fine-tuned</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0.1, 0.1)</td>
<td>Laftr</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000 ($\sigma = 0.001$)</td>
</tr>
<tr>
<td></td>
<td>Cfair</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000 ($\sigma = 0.003$)</td>
</tr>
<tr>
<td></td>
<td>Ffvae</td>
<td>0.9574</td>
<td>0.9787</td>
<td>1.000 ($\sigma = 0.1002$)</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td><strong>1.000</strong></td>
<td><strong>0.0744</strong></td>
<td><strong>0.9625 ($\sigma = 0.0205$)</strong></td>
</tr>
<tr>
<td>(0.9, 0.1)</td>
<td>Laftr</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000 ($\sigma = 0.006$)</td>
</tr>
<tr>
<td></td>
<td>Cfair</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000 ($\sigma = 0.005$)</td>
</tr>
<tr>
<td></td>
<td>Ffvae</td>
<td><strong>0.8723</strong></td>
<td><strong>0.8723</strong></td>
<td>1.000 ($\sigma = 0.092$)</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>0.1383</td>
<td>1.000</td>
<td>1.000 ($\sigma = 0.006$)</td>
</tr>
</tbody>
</table>

(b) **3DShapes dataset:** Our method is able to outperform others in the under-represented case, but the highly correlated scenario of $\text{ci-ratio}=(0.9,0.1)$ is a failure case.
Multiple targets / sensitive attributes

\[
\max_k \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \rho^t_k - \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \rho^s_k
\]
Remaining challenges

- MMD-B-Fair:
  - When $s$ and $t$ are very correlated
  - For attributes with many values (use HSIC?)
Remaining challenges

- MMD-B-Fair:
  - When $s$ and $t$ are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
Remaining challenges

- MMD-B-Fair:
  - When $s$ and $t$ are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
- When $\mathbb{P} \neq \mathbb{Q}$, can we tell how they're different?
Remaining challenges

- **MMD-B-Fair:**
  - When $s$ and $t$ are very correlated
  - For attributes with many values (use HSIC?)

- **Meta-testing:** more powerful approaches, better analysis

- **When $\mathbb{P} \neq \mathbb{Q}$, can we tell how they're different?**
  - Methods so far: low-$d$, and/or points w/ large critic value
Remaining challenges

- MMD-B-Fair:
  - When $s$ and $t$ are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
- When $P \not= Q$, can we tell how they're different?
  - Methods so far: low-$d$, and/or points w/ large critic value
Remaining challenges

- MMD-B-Fair:
  - When $s$ and $t$ are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
- When $P \neq Q$, can we tell how they're different?
  - Methods so far: low-$d$, and/or points w/ large critic value
- Avoid the need for data splitting (selective inference)
Remaining challenges

- MMD-B-Fair:
  - When $s$ and $t$ are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
- When $\mathbb{P} \neq \mathbb{Q}$, can we tell *how* they're different?
  - Methods so far: low-$d$, and/or points with large critic value
- Avoid the need for data splitting (selective inference)
  - Kübler+ NeurIPS-20 gave one method, but very limited
A good takeaway

Combining a deep architecture with a kernel machine that takes the higher-level learned representation as input can be quite powerful.