Say we observe two different datasets:

\[ X \sim P \text{ (model of [6])} \quad Y \sim Q \text{ (MNIST samples)} \]

Our question: is \( P \neq Q \)?

- Did my generative model actually learn the distribution I wanted it to?
- Do smokers and non-smokers have different distributions of cancers?
- Do these neurons fire differently when the subject is looking at image A instead of B?
- Are these different data sources the same? We want to be able to detect any possible difference, without making parametric assumptions, on high-dimensional data.

**Maximum mean discrepancy**

Distance between distributions [2] based on a kernel on sample points \( k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R} \):

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

Estimate the MMD by taking sample means.

**MMD tests**

- Estimate \( \text{MMD}(P, Q) \) with \( \hat{\text{MMD}}(X, Y) \).
- Estimate a threshold \( \hat{c}_m \).
- Shuffle up \( X \cup Y \) into random halves many times; take \( \hat{c}_m \) as the \( 1 - \alpha \) quantile of the \( \hat{\text{MMD}}^2 \)’s.
- Say \( P \neq Q \) if \( m \hat{\text{MMD}}^2(X, Y) > \hat{c}_m \).

**Train-test splits**

Choose a kernel \( k \) in MMD test

\[ X \quad \hat{t} \quad Y \]

\[ \text{Choose a kernel } k \text{ in MMD test} \]

\[ \text{Efficient permutation tests} \]

- Current ways to compute permutations very slow.
- Inefficient memory access pattern.
- Wrote a cache-aware implementation in Shogun.
- 15-30x the speed of existing implementations.
- Faster, more scalable than spectral approximations.

**Model criticism**

- [6]’s MNIST GAN is really good (top-left).
- Can we tell the distributions apart? Yes!
- Gaussian-ARD kernel: \( p \)-values almost exactly 0.
- Pixel weights (right) show where the model’s distribution differs.
- Just optimizing bandwidth: 57% power at \( \alpha = .01 \).
- Median heuristic: 42% power.
- Looking at points with high/low witness function values from ARD kernel (like [5]) gives more insight:
- Model underproduces vertical 1s,
- Overproduces right-slanted digits.
- MMD value very small, but very consistent.

**Optimizing MMD test power**

- When \( P \neq Q \), \( \hat{\text{MMD}}^2 \) is asymptotically normal:
  \[ \hat{\text{MMD}}^2(X, Y) - \text{MMD}^2(P, Q) \sim \mathcal{N}(0,1). \]
- Then test power \( \hat{P} \left( m \hat{\text{MMD}}^2(X, Y) > \hat{c}_m \right) \) goes to
  \[ \Phi \left( \frac{\text{MMD}^2(P, Q)}{\sqrt{V_m(P, Q)} - c_m} \right). \]
  \[ V_m = O(m^{-1}); \text{ MMD, } c_m \text{ are constant in } m. \]
- So, maximize \( \hat{t} = \hat{\text{MMD}}^2(X, Y)/\sqrt{V_m(X, Y)} \).
- \( \hat{V}_m \): quadratic-time, unbiased estimator of \( V_m \).
- Maximize kernel parameters with backprop.

**As a GAN objective**

- Discriminators in standard GANs [3] look at one sample at a time.
- Problem: generator incentivized to produce just one sample that the discriminator likes, then gets stuck.
- Generator distribution should match true one.
- Use a two-sample test as the discriminator!
- [1, 4]: doing this by maximizing the MMD.
- Generative Moment Matching Network (GMMN)
- Instead, optimize \( \hat{t} \) criterion (t-GMMN).
- Or, do distributional feature matching (like [6]):
  - Train discriminator normally.
- Generator uses \( \hat{t} \) with kernel from discriminator.

- Used sum of Gaussian kernels:
  - Not a great kernel on MNIST pixels.
  - Nearly useless on natural image pixels.
  - Gradients decay too fast.
- We’re trying out better kernels.

**References**


