A Mean-Field Theory for Kernel Learning in Generative and Discriminative Models of Machine Learning

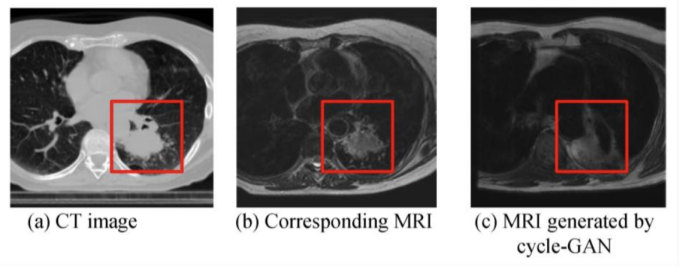
Masoud Badiei Khuzani

November, 2019

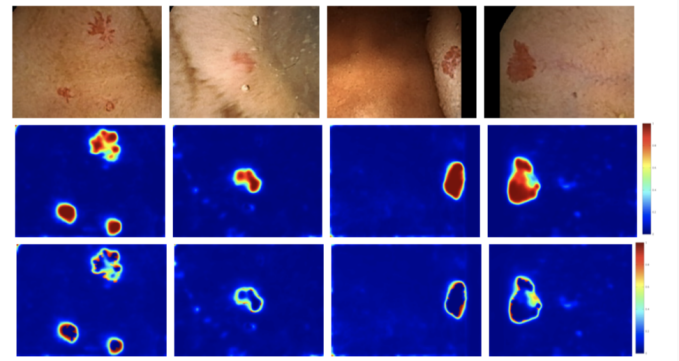
Stanford University

Machine Learning Applications in Medical Imaging

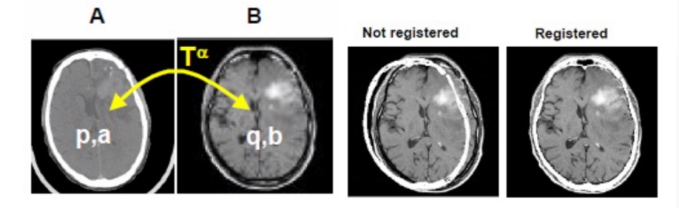
- Machine learning and AI techniques have changed the landscape of medical imaging.
- Many medical tasks such as segmentation, registration, or diagnosis can be done automatically without any intervention from clinicians.
  1. **Modality Transformation**



  2. **Segmentation**



  3. **Registration**



Machine Learning Problems in a Nutshell

-A broad range of machine learning tasks can be reduced to the problem of learning a target function

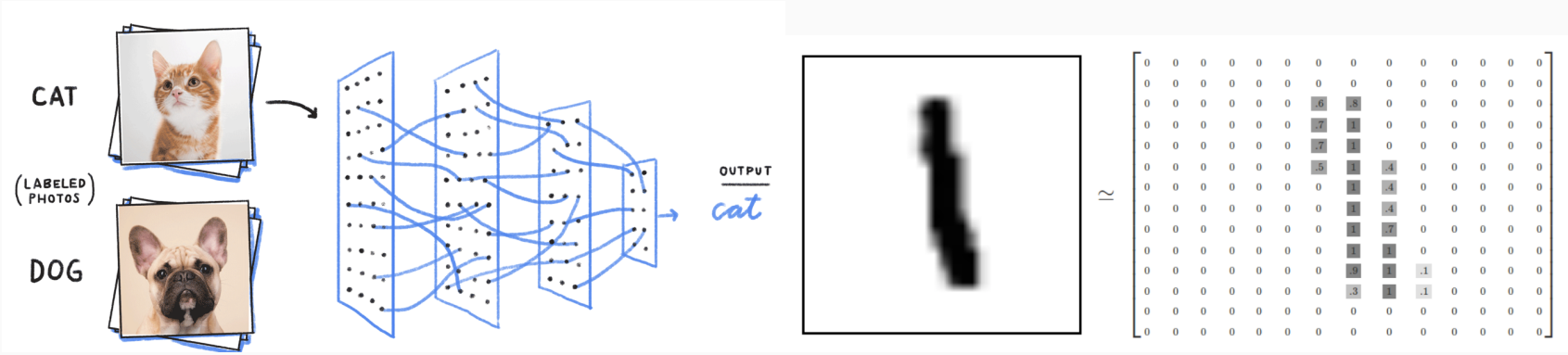
$$h : \mathcal{X} \to \mathcal{Y}.$$

Machine Learning Problems in a Nutshell

-A broad range of machine learning tasks can be reduced to the problem of learning a target function

$$h : \mathcal{X} \to \mathcal{Y}.$$

- **Discriminative (classification) model**: $\mathcal{X} = S^{n\times n} \times S^{n\times n} \times S^{n\times n}, \mathcal{Y} = \{0, 1\}$, $\mathcal{S} = \{0, 1, \cdots, 255\}$.

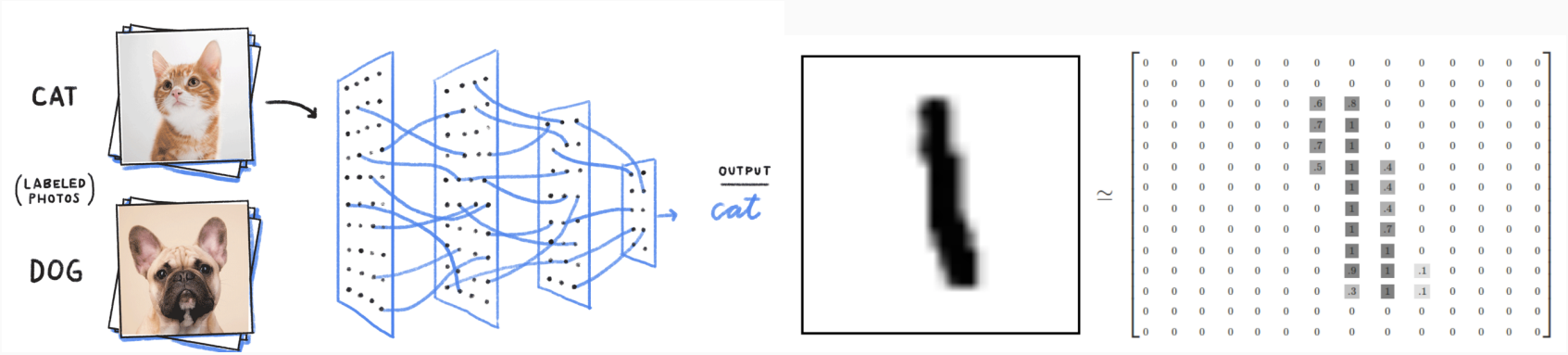


Machine Learning Problems in a Nutshell

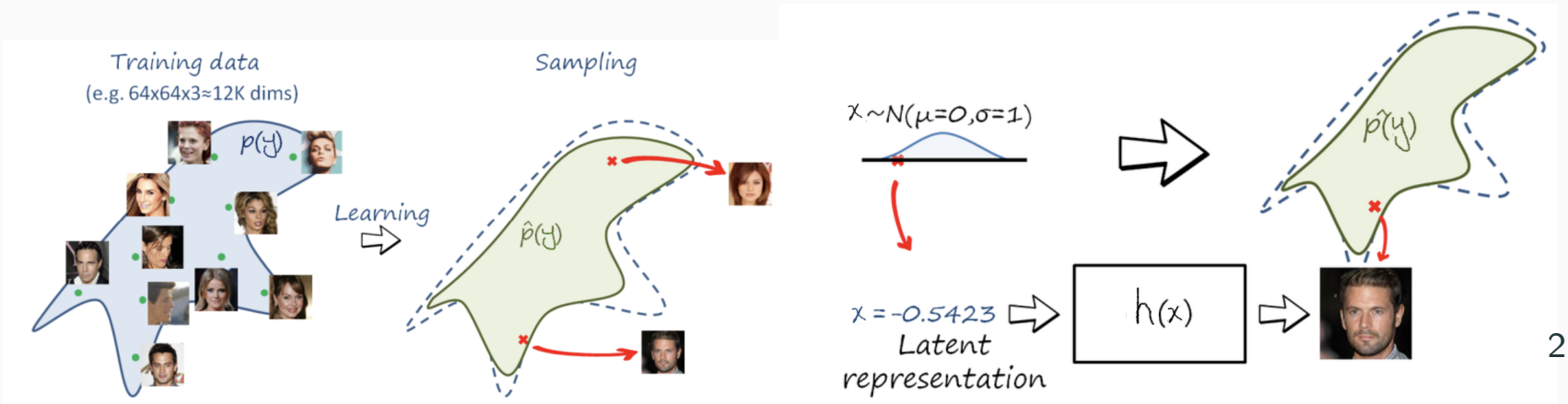
-A broad range of machine learning tasks can be reduced to the problem of learning a target function

$$h : \mathcal{X} \to \mathcal{Y}.$$

- **Discriminative (classification) model**: $\mathcal{X} = S^{n\times n} \times S^{n\times n} \times S^{n\times n}, \mathcal{Y} = \{0, 1\}$, $\mathcal{S} = \{0, 1, \cdots, 255\}$.



- **Generative (sampling) model**: $\mathcal{X} = \mathbb{R}$, $\mathcal{Y} = S^{n\times n} \times S^{n\times n} \times S^{n\times n}$



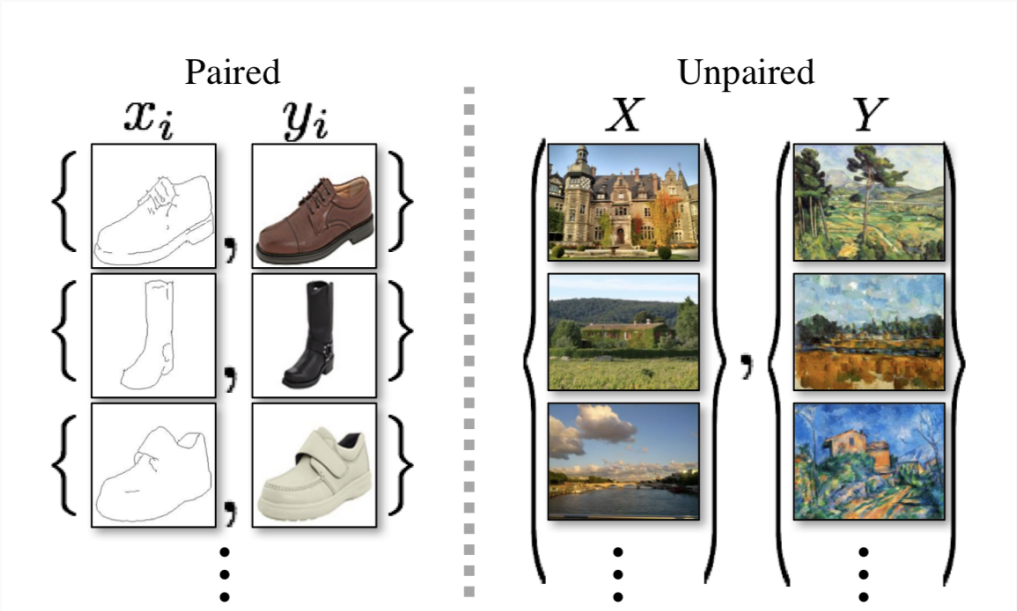
Supervised vs Unsupervised Models

- **Supervised models**: The paired instances from the target map are available

$$(x_1, y_1), \cdots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}, \quad y_i = h(x_i). \tag{1}$$

- **Unsupervised models**: The unpaired instances from each domain is available (for classification it means class label is unavailable)

$$x_1, \cdots, x_n \in \mathcal{X}, \quad y_1, \cdots, y_m \in \mathcal{Y}, \quad y_i \neq h(x_i).$$



How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$ *

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), f(x_i)).$$

  $\mathcal{F}$: a function class, ℓ : a loss function.

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), f(x_i)).$$

  $\mathcal{F}$: a function class, ℓ : a loss function.

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{f \in \mathcal{F}} D\left(\widehat{P}^n = \frac{1}{n} \sum_{i=1}^{n} \delta(y - f(x_i)) || \widehat{Q}^m = \frac{1}{m} \sum_{i=1}^{m} \delta(y - y_i)\right),$$

  $D(\cdot, \cdot)$: divergence between two distributions, $\delta(\cdot)$: Dirac's delta function.

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$ *

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), f(x_i)).$$

  $\mathcal{F}$: a function class, ℓ : a loss function.

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{f \in \mathcal{F}} D\left(\widehat{P}^n = \frac{1}{n} \sum_{i=1}^{n} \delta(y - f(x_i)) || \widehat{Q}^m = \frac{1}{m} \sum_{i=1}^{m} \delta(y - y_i)\right),$$

  $D(\cdot, \cdot)$: divergence between two distributions, $\delta(\cdot)$: Dirac's delta function.

  - The variational form of the distance between two distributions P, Q defined on $\mathcal{X}$:

$$D(P||Q) = \max_{g \in \mathcal{G}} \left| \int_{\mathcal{X}} g(x) \mathrm{d}P(x) - \int_{\mathcal{X}} g(x) \mathrm{d}Q(x) \right|. \tag{2}$$

  $\mathcal{G}$: function class.

Variational form of the divergence

- **Total Variation Distance**:

$$\mathrm{TV}(P||Q) = \sup_{g:\mathcal{X}\to\mathbb{R}:\|g\|_\infty\leq 1/2} \left| \int_{\mathcal{X}} g(x)\mathrm{d}P(x) - \int_{\mathcal{X}} g(x)\mathrm{d}Q(x) \right|.$$

- **Wasserstein (KantorovichRubinstein) Distance**:

$$\mathrm{W}_1(P||Q) = \sup_{g:\mathcal{X}\to\mathbb{R}:\mathrm{Lip}(g)\leq 1} \left| \int_{\mathcal{X}} g(x)\mathrm{d}P(x) - \int_{\mathcal{X}} g(x)\mathrm{d}Q(x) \right|.$$

- **Maximum Mean Discrepancy Distance**:

$$\mathrm{MMD}_{\mathcal{H}}(P||Q) = \sup_{g:\mathcal{X}\to\mathbb{R}:\|g\|_{\mathcal{H}}\leq 1} \left| \int_{\mathcal{X}} g(x)\mathrm{d}P(x) - \int_{\mathcal{X}} g(x)\mathrm{d}Q(x) \right|.$$

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), f(x_i)).$$

$\mathcal{F}$: a function class, ℓ : a loss function.

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

Plug

$$\min_{f \in \mathcal{F}} D\left(\widehat{P}^n = \frac{1}{n} \sum_{i=1}^{n} \delta(y - f(x_i)) || \widehat{Q}^m = \frac{1}{m} \sum_{i=1}^{m} \delta(y - y_i)\right),$$

$D(\cdot, \cdot)$: divergence between two distributions, $\delta(\cdot)$: Dirac's delta function.

- The variational form of the distance between two distributions P, Q defined on $\mathcal{X}$:

$$D(P||Q) = \max_{g \in \mathcal{G}} \left| \int_{\mathcal{X}} g(x) \mathrm{d}P(x) - \int_{\mathcal{X}} g(x) \mathrm{d}Q(x) \right|. \tag{3}$$

$\mathcal{G}$: function class.

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$?

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), f(x_i)).$$

  $\mathcal{F}$: a function class, ℓ : a loss function.

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{f \in \mathcal{F}} \max_{g \in \mathcal{G}} \left| \frac{1}{m} \sum_{i=1}^{m} g(y_i) - \frac{1}{n} \sum_{i=1}^{n} g(f(x_i)) \right|.$$

  $\mathcal{G}$: function class.

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$?

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), f(x_i)).$$

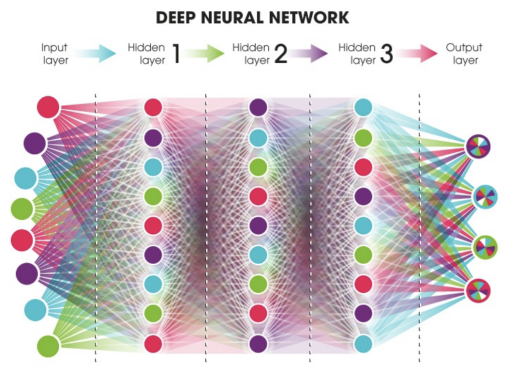
  $\mathcal{F}$: a function class, ℓ : a loss function.

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{f \in \mathcal{F}} \max_{g \in \mathcal{G}} \left| \frac{1}{m} \sum_{i=1}^{m} g(y_i) - \frac{1}{n} \sum_{i=1}^{n} g(f(x_i)) \right|.$$

  $\mathcal{G}$: function class.

Deep Neural Networks



-In deep neural networks with n-layers, the parametric function takes the following form

$$\mathrm{NN}(x; \boldsymbol{W}) \stackrel{\text{def}}{=} \sigma(W_n \sigma(\cdots \sigma(W_1 x))), \quad W_k \in \mathbb{R}^{m_k \times m_{k-1}},$$

where $\boldsymbol{W} \stackrel{\text{def}}{=} (W_1, \cdots, W_n)$.

-$\sigma(\cdot)$ is the activation function for non-linear function approximation.

1. ReLU:$\sigma(x) = \max\{0, x\}$.
2. logisitc: $\sigma(x) = \frac{1}{1+e^{-x}}$.
3. arctan: $\sigma(x) = \tan^{-1}(x)$.

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$?

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)).$$

  $\mathcal{F}$: a function class, ℓ : a loss function.

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{f \in \mathcal{F}} \max_{g \in \mathcal{G}} \left| \frac{1}{m} \sum_{i=1}^{m} g(y_i) - \frac{1}{n} \sum_{i=1}^{n} g(f(x_i)) \right|.$$

  $\mathcal{G}$: function class.

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$?

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{W_1,\cdots,W_n} \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \mathrm{NN}(x_i, \boldsymbol{W})) + \sum_{i=1}^{n} \frac{\lambda_i}{2}\|W_i\|_F^2.$$

$$\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y}, x \mapsto f(x) = \mathrm{NN}(x_i, \boldsymbol{W}), \boldsymbol{W} \in \textstyle\prod_{k=1}^{n} \mathbb{R}^{m_k \times m_{k-1}}\}.$$

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{W_1,\cdots,W_n} \max_{W_1',\cdots,W_n'} \left| \frac{1}{m}\sum_{i=1}^{m} \mathrm{NN}(\boldsymbol{W}', y_i) - \frac{1}{n}\sum_{i=1}^{n} \mathrm{NN}(\boldsymbol{W}', \mathrm{NN}(\boldsymbol{W}, \boldsymbol{x})) \right|.$$

$$\mathcal{G} = \{g : \mathcal{Y} \to \mathbb{R}, x \mapsto g(y) = \mathrm{NN}(y, \boldsymbol{W}), \boldsymbol{W} \in \textstyle\prod_{k=1}^{n} \mathbb{R}^{m_k \times m_{k-1}}\}.$$

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$?

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

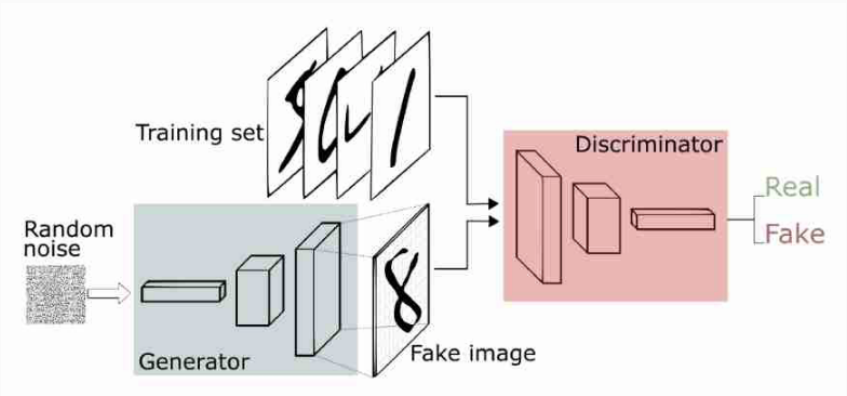
$$\min_{W_1,\cdots,W_n} \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \mathrm{NN}(x_i, \boldsymbol{W})) + \sum_{i=1}^{n} \frac{\lambda_i}{2}\|W_i\|_F^2.$$

$$\mathcal{F} = \{f : \mathcal{X} \to \mathcal{Y}, x \mapsto f(x) = \mathrm{NN}(x_i, \boldsymbol{W}), \boldsymbol{W} \in \textstyle\prod_{k=1}^{n} \mathbb{R}^{m_k \times m_{k-1}}\}.$$

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

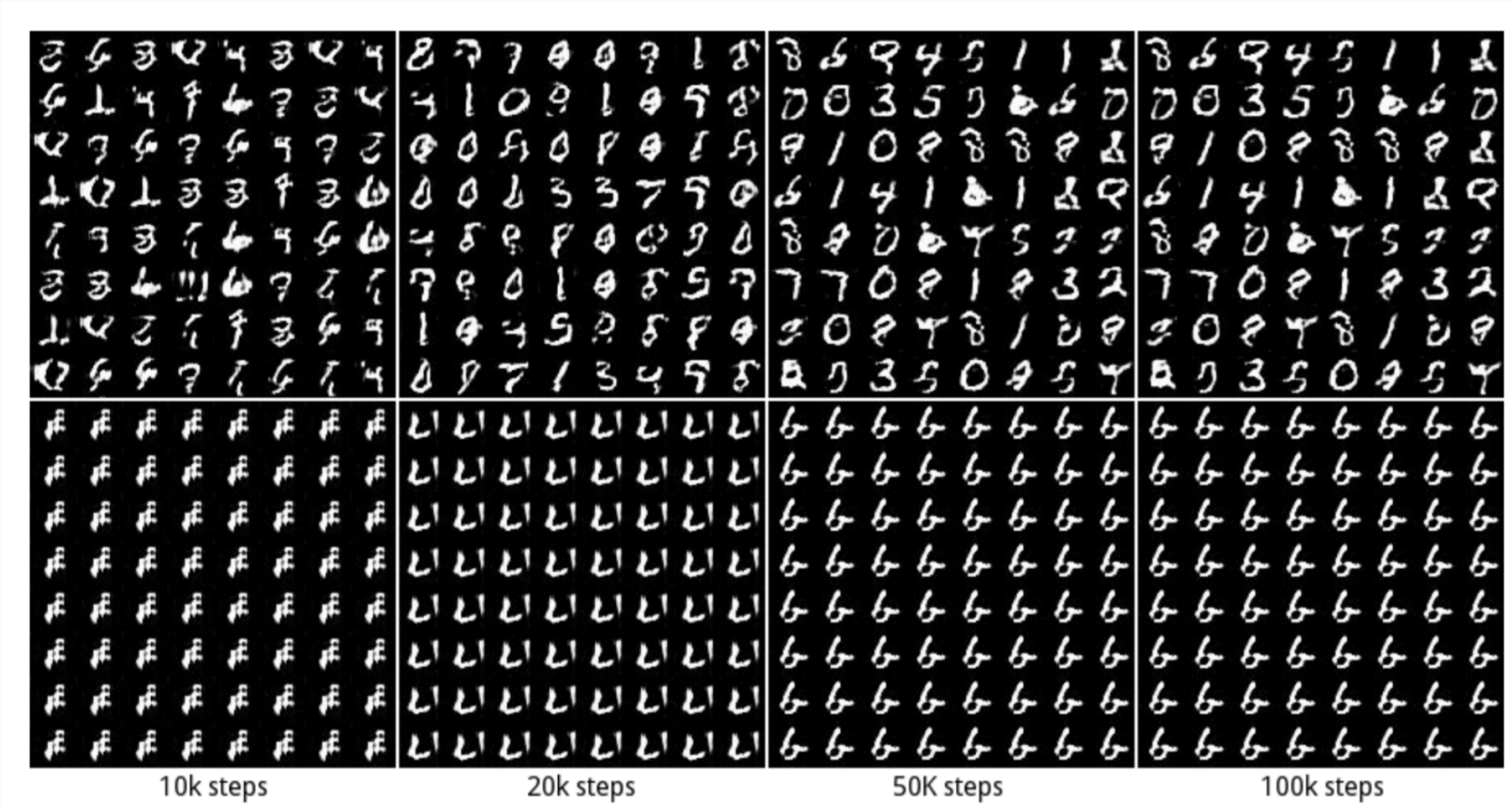
$$\min_{W_1,\cdots,W_n} \max_{W_1',\cdots,W_n'} \underbrace{\left| \frac{1}{m}\sum_{i=1}^{m} \mathrm{NN}(\boldsymbol{W}', y_i) - \frac{1}{n}\sum_{i=1}^{n} \mathrm{NN}(\boldsymbol{W}', \mathrm{NN}(\boldsymbol{W}, \boldsymbol{x})) \right|}_{\text{Generative Adverserial Networks}}.$$

$$\mathcal{G} = \{g : \mathcal{Y} \to \mathbb{R}, x \mapsto g(y) = \mathrm{NN}(y, \boldsymbol{W}), \boldsymbol{W} \in \textstyle\prod_{k=1}^{n} \mathbb{R}^{m_k \times m_{k-1}}\}.$$



Training Issues in Deep Neural Networks

- Training GANs with DNNs is difficult, and often results in the mode collapse.



Kernel Machines

- Let $\mathcal{H}$ denotes the Hilbert space (inner product space with Cauchy sequence limits) of real-valued functions on $\mathcal{X}$.
- For $x \in \mathcal{X}$, consider the map $L_x : \mathcal{H} \to \mathbb{R}, f \mapsto L_x[f] = f(x)$. If L_x is a bounded operator, we say $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS).
- $L_x \in \mathcal{H}^*$, where $\mathcal{H}^*$ is the dual-space of the Hilbert space H.
- By Riesz representation theorem, there exists an element $\phi(x) \in \mathcal{H}$, such that

$$L_x(f) = \langle f, \phi(x) \rangle_{\mathcal{H}}, \quad \forall f \in \mathcal{H}.$$


David Hilbert

- In particular, $\phi : \mathcal{X} \to \mathcal{H}$ and $\phi(y) \in \mathcal{H}$. Therefore,

$$L_x(\phi(y)) = \langle \phi(y), \phi(x) \rangle_{\mathcal{H}} \stackrel{\text{def}}{=} K(x, y).$$

- $K(x, y)$: kernel function, $\phi(x)$: feature map.

Kernel Machines

Moore-Aronszajn Theorem . Consider the kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is symmetric and is positive definite in the sense that

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j K(x_i, x_j) > 0,$$

for all $n \in \mathbb{N}$, $c_1, \cdots, c_n \in \mathbb{R}$, and $x_1, \cdots, x_n \in \mathcal{X}$. Then, there exists a unique Hilbert space $\mathcal{H}_K$ for which K has the reproducing property. Furthermore, for every function $f \in \mathcal{H}_K$, we have the following expansion

$$f(x) = \sum_{i=1}^{\infty} w_i K(x, x_i), \quad \text{for some } x_1, x_2, \cdots \in \mathcal{X}.$$

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$?

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(x_i)) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(x_i), f(x_i)).$$

  $\mathcal{F}$: a function class, ℓ : a loss function.

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{f \in \mathcal{F}} \max_{g \in \mathcal{G}} \left| \frac{1}{m} \sum_{i=1}^{m} g(y_i) - \frac{1}{n} \sum_{i=1}^{n} g(f(x_i)) \right|.$$

  $\mathcal{G}$: function class.

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$?

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set

$$\min_{(w_1,\cdots,w_n)\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^{n} \ell\Big(y_i, \sum_{j=1}^{n} w_j K(x_i, x_j)\Big) + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2.$$

$\mathcal{F} = \mathcal{H}_K = \left\{f : \mathcal{X} \to \mathcal{Y}, x \mapsto \sum_{i=1}^{n} w_i K(x, x_i), \boldsymbol{w} \in \mathbb{R}^n\right\}$

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{W_1,\cdots,W_n} \max_{\|g\|_{\mathcal{H}_K}\leq 1} \left| \frac{1}{m}\sum_{i=1}^{m} g(y_i) - \frac{1}{n}\sum_{i=1}^{n} g(\mathrm{NN}(\boldsymbol{W}, x_i)) \right|.$$

$\mathcal{G} = \{g : \mathcal{X} \to \mathbb{R} : \|g\|_{\mathcal{H}_K} \leq 1\}.$

How to Learn The Target Map $h : \mathcal{X} \to \mathcal{Y}$?

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set ($\ell(y, x) = \max\{0, 1 - xy\}$.)

$$\min_{(w_1, \cdots, w_n) \in \mathbb{R}^n} \underbrace{\frac{1}{n} \sum_{i=1}^{n} \ell\Big(y_i, \sum_{j=1}^{n} w_j K(x_i, x_j)\Big) + \frac{\lambda}{2} \|\boldsymbol{w}\|_2^2}_{\text{Kernel Support Vector Machines}}.$$

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{W_1, \cdots, W_n} \mathrm{MMD}_K(\widehat{P}^n_{\boldsymbol{W}} || \widehat{Q}^m) = \frac{1}{m^2} \sum_{i,j=1}^{m} K(y_i, y_j) + \frac{1}{n^2} \sum_{i,j=1}^{n} K(\mathrm{NN}(\boldsymbol{W}, x_i), \mathrm{NN}(\boldsymbol{W}, x_i))$$
$$\underbrace{- \frac{2}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} K(y_j, \mathrm{NN}(\boldsymbol{W}, x_i))}_{\text{Generative Moment Matching Networks}},$$

, where $y_1, \cdots, y_m \sim_{\text{i.i.d.}} Q$, and $\mathrm{NN}(\boldsymbol{W}, x_1), \cdots, \mathrm{NN}(\boldsymbol{W}, x_n) \sim_{\text{i.i.d.}} P_{\boldsymbol{W}}$.

Kernel Selection Problem

Kernel Model Selection Problem: A bad kernel may yield a poor machine learning system.

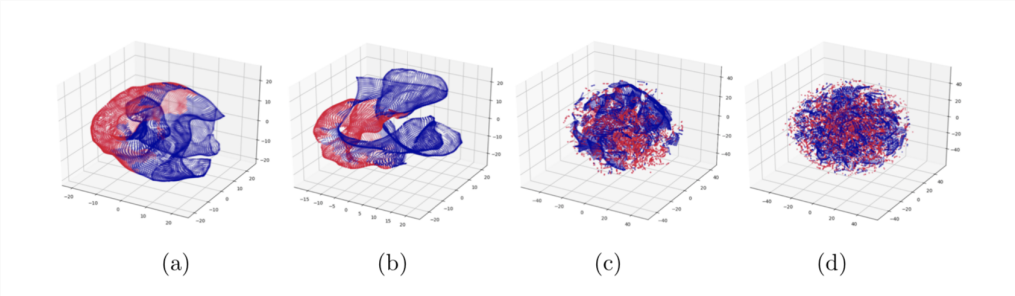
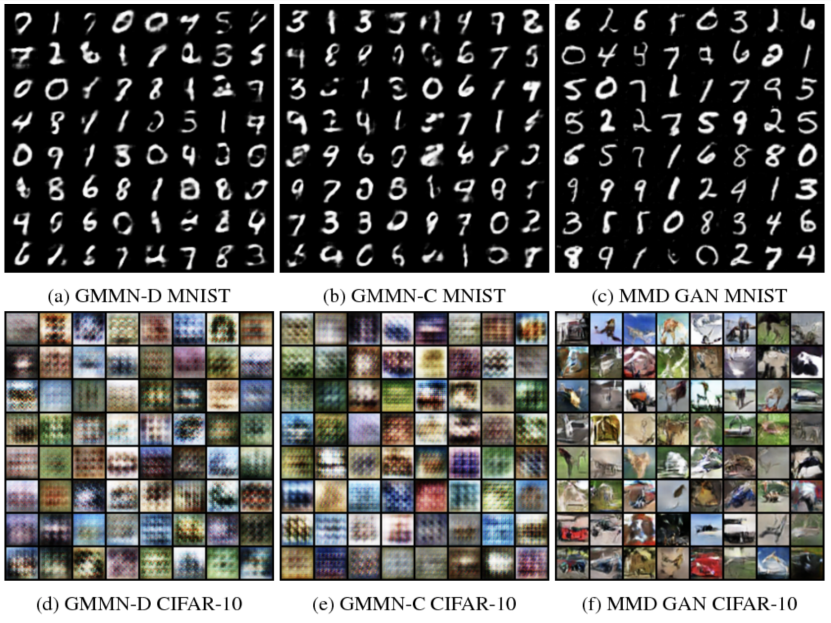


Figure 1: t-SNE plot for feature maps generated by Gaussian kernel $K(x, y) = e^{-\gamma\|x-y\|_2}$ for different bandwidth values $\gamma > 0$.



Kernel Optimization

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set ($\ell(y,x) = \max\{0, 1-xy\}$.)

$$\min_{(w_1,\cdots,w_n)\in\mathbb{R}^n} \max_{K\in\mathcal{K}} \underbrace{\frac{1}{n}\sum_{i=1}^{n} \ell\Big(y_i, \sum_{j=1}^{n} w_j K(x_i,x_j)\Big) + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2}_{\text{Kernel Support Vector Machines}}.$$

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\min_{W_1,\cdots,W_n} \max_{K\in\mathcal{K}} \mathrm{MMD}_K(\widehat{P}^n_{\boldsymbol{W}} || \widehat{Q}^m) = \frac{1}{m^2}\sum_{i,j=1}^{m} K(y_i,y_j) + \frac{1}{n^2}\sum_{i,j=1}^{n} K(\mathrm{NN}(\boldsymbol{W},x_i),\mathrm{NN}(\boldsymbol{W},x_i))$$
$$\underbrace{- \frac{2}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} K(y_j,\mathrm{NN}(\boldsymbol{W},x_i))}_{\text{Generative Moment Matching Networks}}$$

, where $y_1,\cdots,y_m \sim_{\text{i.i.d.}} Q$, and $\mathrm{NN}(\boldsymbol{W},x_1),\cdots,\mathrm{NN}(\boldsymbol{W},x_n) \sim_{\text{i.i.d.}} P_{\boldsymbol{W}}$.

Kernel Optimization

- Suppose the kernel function is shift invariant $K(\boldsymbol{x}, \boldsymbol{y}) = K(\boldsymbol{x} - \boldsymbol{y})$.
- Rahimi and Recht [1] proved that shift invariant kernels have the following Fourier representation

$$K(x - y) = \mathbb{E}_{\mu}[\varphi(x, \boldsymbol{\xi})\varphi(y, \boldsymbol{\xi})] = \int_{\mathbb{R}^D} \varphi(x, \boldsymbol{\xi})\varphi(y, \boldsymbol{\xi}) \mathrm{d}\mu(\boldsymbol{\xi}),$$

  where $\varphi(\boldsymbol{x}, \boldsymbol{\xi}) = \cos(\langle \boldsymbol{x}, \boldsymbol{\xi} \rangle + b)$, $b \sim \mathrm{Uniform}([0, 2\pi])$.
- $\varphi(x, \xi)$ is called the random feature map since

$$K(x - y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}} = \langle \varphi(x, \xi), \varphi(y, \xi) \rangle_{L^2(\mu)}.$$

[1] Ali Rahimi and Benjamin Recht. *Random Features for Large-Scale Kernel Machines* NIPS 2007.

Kernel Optimization

- **Supervised models**: the target map is learned by minimizing a risk function over a given training set ($\ell(y,x) = \max\{0, 1-xy\}$.)

$$\min_{(w_1,\cdots,w_n)\in\mathbb{R}^n} \max_{\mu\in\mathcal{P}} \underbrace{\frac{1}{n}\sum_{i=1}^{n} \ell\Big(y_i, \sum_{j=1}^{n} w_j \mathbb{E}_\mu[\varphi(x_i,\boldsymbol{\xi})\varphi(x_j,\boldsymbol{\xi})]\Big) + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2}_{\text{Kernel Support Vector Machines}}.$$

- **Unsupervised models**: the target map is learned by minimizing the distance between the empirical distribution of data and the model output

$$\begin{aligned}\min_{W_1,\cdots,W_n} \max_{\mu\in\mathcal{P}} \mathrm{MMD}_K(\widehat{P}^n_{\boldsymbol{W}} || \widehat{Q}^m) = &\frac{1}{m^2}\sum_{i,j=1}^{m} \mathbb{E}_\mu[\varphi(y_i,\boldsymbol{\xi})\varphi(y_j,\boldsymbol{\xi})] \\ &+ \frac{1}{n^2}\sum_{i,j=1}^{n} \mathbb{E}_\mu[\varphi(\mathrm{NN}(\boldsymbol{W},x_i),\boldsymbol{\xi})\varphi(\mathrm{NN}(\boldsymbol{W},x_i),\boldsymbol{\xi})] \\ &- \frac{2}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} \mathbb{E}_\mu[\varphi(y_j,\boldsymbol{\xi})\varphi(\mathrm{NN}(\boldsymbol{W},x_i),\boldsymbol{\xi})].\end{aligned}$$

, where $y_1,\cdots,y_m \sim_{\text{i.i.d.}} Q$, and $\mathrm{NN}(\boldsymbol{W},x_1),\cdots,\mathrm{NN}(\boldsymbol{W},x_n) \sim_{\text{i.i.d.}} P_{\boldsymbol{W}}$.

Kernel Optimization

- **Supervised Models**: For binary classification $\mathcal{Y} = \{-1, +1\}$ using kernel SVMs, we optimize the kernel target alignment.

$$\max_{\mu \in \mathcal{P}} \frac{2}{n(n-1)} \sum_{1 \le i < j \le n} y_i y_j \mathbb{E}_\mu[\varphi(x_i, \boldsymbol{\xi})\varphi(x_j, \boldsymbol{\xi})].$$

- **Unsupervised Models**: We optimize the kernel target alignment ($M = m + n$)

$$\min_{W_1, \cdots, W_n} \max_{\mu \in \mathcal{P}} \frac{2}{M(M-1)} \sum_{1 \le i < j \le n+m} z_i z_j \mathbb{E}_\mu[\varphi(v_i, \boldsymbol{\xi})\varphi(v_j, \boldsymbol{\xi})],$$

where $z_1, \cdots, z_M \sim \mathrm{Uniform}\{-1, +1\}$ and

1. If $z_i = +1$, let $v_i \in \{y_1, \cdots, y_m\}$ (true data).
2. If $z_i = -1$ let $v_i \in \{\mathrm{NN}(\boldsymbol{W}, x_1), \cdots, \mathrm{NN}(\boldsymbol{W}, x_n)\}$ (generated data).

Kernel Optimization

- **Step 1**: Lets focus on the following optimization problem

$$\min_{W_1,\cdots,W_n} \max_{\mu\in\mathcal{P}} \frac{2}{M(M-1)} \sum_{1\le i<j\le n+m} z_i z_j \mathbb{E}_\mu[\varphi(v_i,\boldsymbol{\xi})\varphi(v_j,\boldsymbol{\xi})],$$

  where $z_1,\cdots,z_M \sim \mathrm{Uniform}\{-1,+1\}$ and

  1. If $z_i = +1$, let $v_i \in \{y_1,\cdots,y_m\}$.
  2. If $z_i = -1$ let $v_i \in \{\mathrm{NN}(\boldsymbol{W},x_1),\cdots,\mathrm{NN}(\boldsymbol{W},x_n)\}$.

- We rewrite the problem as a risk minimization

$$\min_{W_1,\cdots,W_n} \max_{\mu\in\mathcal{P}} \frac{2}{\alpha M(M-1)} \sum_{1\le i<j\le n} (\alpha z_i z_j - \mathbb{E}_\mu[\varphi(v_i,\boldsymbol{\xi})\varphi(v_j,\boldsymbol{\xi})])^2.$$

Kernel Optimization

- **Step 2**: We apply the Monte-Carlo sample average approximation.
- In particular, we optimize with respect to samples of the i.i.d. samples of the target distribution $\boldsymbol{\xi}^1, \cdots, \boldsymbol{\xi}^N \sim_{\text{i.i.d.}} \mu$,

$$\min_{W_1,\cdots,W_n} \max_{\widehat{\mu}^N \in \mathcal{P}^N} \frac{2}{\alpha M(M-1)} \sum_{1 \le i < j \le n} \left(\alpha z_i z_j - \frac{1}{N} \sum_{k=1}^{N} \varphi(x, \boldsymbol{\xi}_k) \varphi(y, \boldsymbol{\xi}_k) \right)^2,$$

  where $\mathcal{P}^N \stackrel{\text{def}}{=} \{\widehat{\mu}^N \in \mathcal{M}(\mathbb{R}^D) : D(\widehat{\mu} || \widehat{\mu}_0^N) \le R\}$.
- We define the empirical distribution of the samples (particles) as below

$$\widehat{\mu}^N(\boldsymbol{\xi}) = \frac{1}{N} \sum_{i=1}^{N} \delta(\boldsymbol{\xi} - \boldsymbol{\xi}^k). \tag{4}$$

Particle Stochastic Gradient Descent

- **Step 3**: Use a particle stochastic gradient descent method to solve the risk minimization as follows:
  1. Initialize the samples $\xi_0^1, \cdots, \xi_0^N \sim_{\text{i.i.d.}} \mu_0$.
  2. At iteration $\ell = 0, 1, 2, \cdots$, we draw two fresh labels $z_\ell, \widetilde{z}_\ell \sim \text{Uniform}\{-1, +1\}$.
  3. Sample $v_\ell \in \{y_1, \cdots, y_m\}$ if $z_\ell = 1$, and $v_\ell \in \{\text{NN}(\boldsymbol{W}, x_1), \cdots, \text{NN}(\boldsymbol{W}, x_n)\}$ $z_\ell = -1$. We pick $\widetilde{v}_t$ using a similar rule.
  4. Apply the particle SGD with the step size $\eta > 0$

$$\boldsymbol{\xi}_{\ell+1}^k = \boldsymbol{\xi}_\ell^k - \frac{\eta}{N}\left(z_\ell \widetilde{z}_\ell - \frac{1}{\alpha N}\sum_{k=1}^N \varphi(\boldsymbol{v}_\ell; \boldsymbol{\xi}_\ell^k)\varphi(\widetilde{\boldsymbol{v}}_\ell; \boldsymbol{\xi}_\ell^k)\right) \nabla_{\boldsymbol{\xi}}\Big(\varphi(\boldsymbol{v}_\ell; \boldsymbol{\xi}_\ell^k)\varphi(\widetilde{\boldsymbol{v}}_\ell; \boldsymbol{\xi}_\ell^k)\Big),$$

     for $k = 1, 2, \cdots, N$.
  5. Approximate the kernel

$$K_{\ell+1}(x, y) \approx \frac{1}{N}\sum_{k=1}^N \varphi(x; \boldsymbol{\xi}_{\ell+1}^k)\varphi(y; \boldsymbol{\xi}_{\ell+1}^k). \tag{5}$$

Evolution of the Histogram of SGD Particles

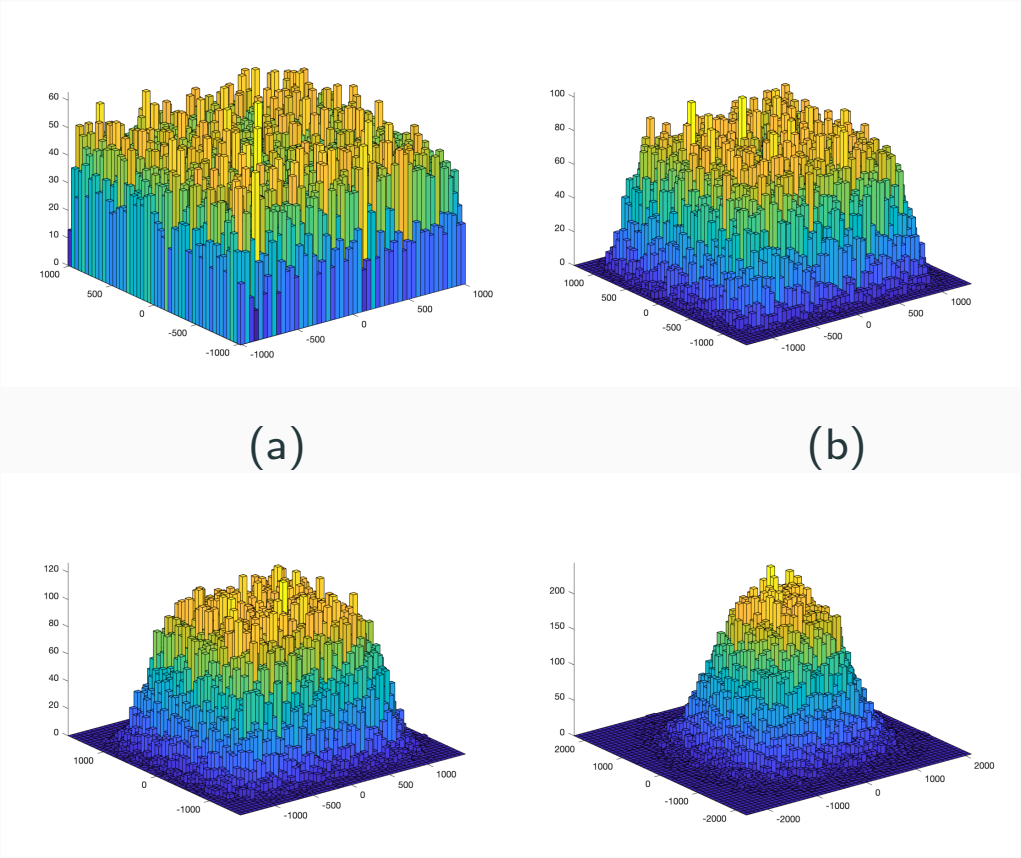


Figure 2: The evolution of the empirical measure $\mu_\ell^N(\boldsymbol{\xi}) = \frac{1}{N}\sum_{k=1}^N \delta(\boldsymbol{\xi} - \boldsymbol{\xi}_\ell^k)$ of the SGD particles $\boldsymbol{\xi}_\ell^1, \cdots, \boldsymbol{\xi}_\ell^N \in \mathbb{R}^2$ at different iterations ℓ. The empirical measure of random feature maps seemingly converges to a Gaussian stationary measure corresponding to a Gaussian RBF kernel. Panel (a): $\ell = 0$, Panel (b): $\ell = 300$, Panel (c): $\ell = 1000$, and Panel (d): $\ell = 2500$.

Consistency of Monte-Carlo Approximations

Theorem: Consider the distribution ball with respect to the 2-Wasserstein distance $\mathcal{P} = \{\mu \in \mathcal{M}(\mathbb{R}^D) : W_2(\mu||\mu_0) \leq R\}$, where μ_0 is a user-defined distribution. Furthermore, consider

$$(\boldsymbol{W}_*, \mu_*) \stackrel{\text{def}}{=} \arg\min_{\boldsymbol{W}\in\mathcal{W}} \arg\sup_{\mu\in\mathcal{P}} \mathrm{MMD}_\mu[P_{\boldsymbol{W}}, Q]$$

$$(\widehat{\boldsymbol{W}}_*^N, \widehat{\mu}_*^N) \stackrel{\text{def}}{=} \arg\min_{\boldsymbol{W}\in\mathcal{W}} \arg\inf_{\widehat{\mu}^N\in\mathcal{P}_N} \widehat{\mathrm{MMD}}^{\alpha}_{\widehat{\mu}^N}[\widehat{P}^n_{\boldsymbol{W}}, \widehat{Q}^n],$$

where

$$\mathrm{MMD}_\mu[P_{\boldsymbol{W}}, Q] = \mathbb{E}_{P^{\otimes 2}}[\mathbb{E}_\mu[\varphi(\mathrm{NN}(\boldsymbol{x}, \boldsymbol{W}), \boldsymbol{\xi})\varphi(\mathrm{NN}(\boldsymbol{x}, \boldsymbol{W}), \boldsymbol{\xi})]] + \mathbb{E}_{Q^{\otimes 2}}[\mathbb{E}_\mu[\varphi(y, \boldsymbol{\xi})\varphi(y', \boldsymbol{\xi})]] - 2\mathbb{E}_{P,Q}[\mathbb{E}_\mu[\varphi(\mathrm{NN}(\boldsymbol{x}, \boldsymbol{W}), \boldsymbol{\xi})\varphi(y, \boldsymbol{\xi})]],$$

and

$$\widehat{\mathrm{MMD}}^{\alpha}_{\widehat{\mu}^N}[\widehat{P}^n_{\boldsymbol{W}}, \widehat{Q}^n] = \frac{2}{\alpha 2n(2n-1)} \sum_{1\leq i<j\leq n} \left(\alpha z_i z_j - \frac{1}{N}\sum_{k=1}^{N} \varphi(\mathrm{NN}(v_i, \boldsymbol{W}), \boldsymbol{\xi}_k)\varphi(\mathrm{NN}(v_j, \boldsymbol{W}), \boldsymbol{\xi}_k)\right)^2.$$

Consistency of Monte-Carlo Approximations

Then, with the probability of (at least) $1 - 3\varrho$ over the training data samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ and the random feature samples $\{\boldsymbol{\xi}_0^k\}_{k=1}^N$, the following non-asymptotic bound holds

$$\left|\mathrm{MMD}_\mu[P_{\boldsymbol{W}}, Q] - \widehat{\mathrm{MMD}}^{\alpha}_{\widehat{\mu}^N}[\widehat{P}^n_{\boldsymbol{W}}, \widehat{Q}^n]\right| \leq \sqrt{\frac{L^2(d+2)}{N}} \ln^{\frac{1}{2}}\left(\frac{2^8 N \mathrm{diam}^2(\mathcal{X})}{\varrho}\right) + \frac{8L^2}{\alpha} + 2\max\left\{\frac{c_1 L^2}{n}\ln^{\frac{1}{2}}\left(\frac{4}{\varrho}\right), \frac{c_2 R L^4}{n^2}\ln\left(\frac{4e^{\frac{L^4}{9}}}{\varrho}\right)\right\},$$

where $c_1 = 3^{\frac{1}{4}} \times 2^4$, and $c_2 = 9 \times 2^{11}$.

Consistency of Monte-Carlo Approximations

Then, with the probability of (at least) $1 - 3\varrho$ over the training data samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ and the random feature samples $\{\boldsymbol{\xi}_0^k\}_{k=1}^N$, the following non-asymptotic bound holds

$$\left|\mathrm{MMD}_\mu[P_{\boldsymbol{W}}, Q] - \widehat{\mathrm{MMD}}_{\widehat{\mu}^N}^{\alpha}[\widehat{P}_{\boldsymbol{W}}^n, \widehat{Q}^n]\right| \leq \sqrt{\frac{L^2(d+2)}{N} \ln^{\frac{1}{2}}\left(\frac{2^8 N \mathrm{diam}^2(\mathcal{X})}{\varrho}\right) + \frac{8L^2}{\alpha}} + 2\max\left\{\frac{c_1 L^2}{n} \ln^{\frac{1}{2}}\left(\frac{4}{\varrho}\right), \frac{c_2 R L^4}{n^2} \ln\left(\frac{4e^{\frac{L^4}{9}}}{\varrho}\right)\right\},$$

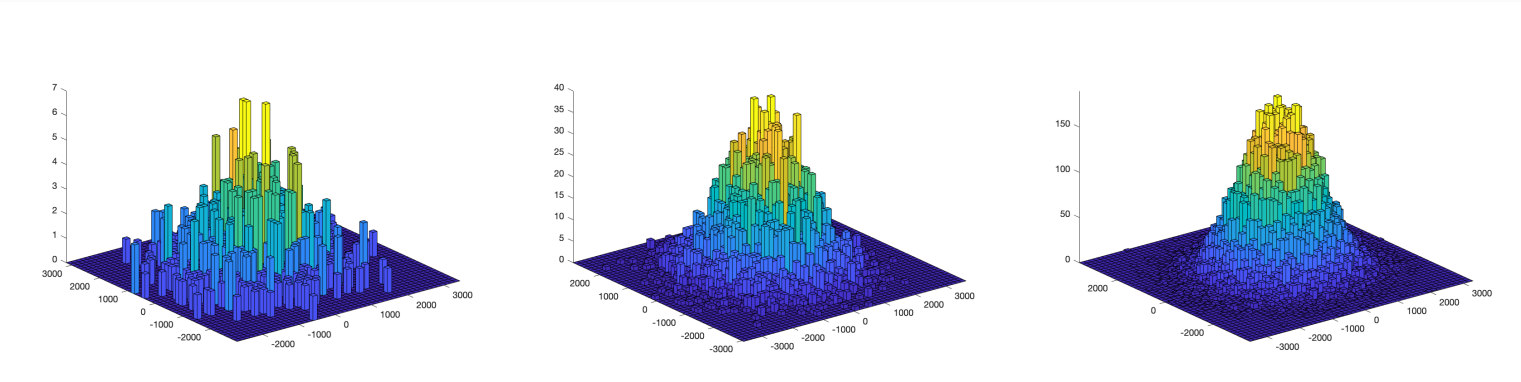
where $c_1 = 3^{\frac{1}{4}} \times 2^4$, and $c_2 = 9 \times 2^{11}$.

N: The number of random feature samples (Particles) $\boldsymbol{\xi}^1, \cdots, \boldsymbol{\xi}^N$ in Particle SGD.

n: The number of training samples $y_1, \cdots, y_n \in \mathcal{Y}$, and $x_1, \cdots, x_n \in \mathcal{X}$.

α: Regularization parameter of the empirical risk minimization.

Convergence of the empirical measure to a limiting measure



(a) (b) (c)

Figure 3: The histogram of the SGD particles at a fixed iteration $\ell = 10000$ and for different number of particles. Panel (a): $N = 1000$, Panel (b): $N = 10000$, Panel (c): $N = 50000$.

- As $N \to \infty$, it seems that $\widehat{\mu}_\ell^N = \frac{1}{N}\sum_{k=1}^N \delta(\boldsymbol{\xi} - \boldsymbol{\xi}_\ell^k)$ converges to a limiting measure μ_ℓ^*.

Consistency of the Particle SGD

Theorem: (McKean-Vlasov Mean-Field PDE) Define the scaled empirical measure of the SGD particles embedded in the continuous time

$$\mu_t^N = \widehat{\mu}_{\lfloor Nt \rfloor}^N = \frac{1}{N}\sum_{k=1}^{N} \delta(\boldsymbol{\xi} - \boldsymbol{\xi}_{\lfloor Nt \rfloor}), \quad 0 \leq t \leq T.$$

Further, suppose that the Lebesgue density of the initial measure of particles $q_0(\boldsymbol{\xi}) = \mu_0(\mathrm{d}\boldsymbol{\xi})/\mathrm{d}\boldsymbol{\xi}$ exists. Then, there exists a unique solution $(p_t^*(\boldsymbol{\xi}))_{0\leq t\leq T}$ to the following non-linear partial differential equation

$$\frac{\partial p_t(\boldsymbol{\xi})}{\partial t} \tag{7}$$
$$= -\frac{\eta}{\alpha}\iint_{\mathcal{X}\times\mathcal{Y}} \left(\int_{\mathbb{R}^p} \varphi(v,\widetilde{\boldsymbol{\xi}})\varphi(\widetilde{v},\widetilde{\boldsymbol{\xi}})p_t(\widetilde{\boldsymbol{\xi}})\mathrm{d}\widetilde{\boldsymbol{\xi}} - \alpha z\widetilde{z}\right)\nabla_{\boldsymbol{\xi}}(p_t(\boldsymbol{\xi})\nabla_{\boldsymbol{\xi}}(\varphi(v;\boldsymbol{\xi})\varphi(\widetilde{v};\boldsymbol{\xi}))\mathrm{d}P_{V,Z}^{\otimes 2},$$

where $P_{V,Z}$ has the marginals $P_{V|Z=+1} = P_{\boldsymbol{W}}$ and $P_{V|Z=-1} = Q$. The PDE is initialized at $p_0(\boldsymbol{\xi}) = q_0(\boldsymbol{\xi})$. Moreover, the measure-valued process $\{(\mu_t^N)_{0\leq t\leq T}\}_{N\in\mathbb{N}}$ converges (weakly) to the unique solution $\mu_t^*(\boldsymbol{\xi}) = p_t^*(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi}$ as the number of particles tend to infinity $N \to \infty$.

Consistency of the Particle SGD

- The Mean-Field PDE can be viewed as the gradient flow for minimizing an energy functional

$$\frac{\mathrm{d}p_t(\boldsymbol{\xi})}{\mathrm{d}t} = -\eta \cdot \mathrm{grad}_{p_t} E_\alpha(p_t(\boldsymbol{\xi})), \quad p_0(\boldsymbol{\xi}) = q_0(\boldsymbol{\xi}),$$

where $\mathrm{grad}_{p_t} E(p_t(\boldsymbol{\xi})) = \nabla_{\boldsymbol{\xi}} \cdot (p_t(\boldsymbol{\xi}) \nabla_{\boldsymbol{\xi}} R_\beta(p_t(\boldsymbol{\xi})))$ is the Riemannian gradient of $R_\beta(\mu_t(\boldsymbol{\xi}))$ with respect to the metric of the Wasserstein manifold, and

$$\inf_{\mu \in \mathcal{M}(\mathbb{R}^p)} E_\alpha(p_t(\boldsymbol{\xi})) \stackrel{\text{def}}{=} \frac{1}{\alpha} \int_{\mathbb{R}^p} R_\alpha(\boldsymbol{\xi}, p_t(\boldsymbol{\xi})) p_t(\boldsymbol{\xi}) \mathrm{d}\boldsymbol{\xi}$$

$$R_\beta(\boldsymbol{\xi}, p_t(\boldsymbol{\xi})) \stackrel{\text{def}}{=} -\alpha (\mathbb{E}_{P_{V,Z}}[Z\varphi(V;\boldsymbol{\xi})])^2 + \mathbb{E}_{\widetilde{\boldsymbol{\xi}} \sim p_t}\left[\left(\mathbb{E}_{P_V}[\varphi(V;\boldsymbol{\xi})\varphi(V;\widetilde{\boldsymbol{\xi}})]\right)^2\right],$$

- The Energy functional is precisely the population MMD, *i.e.*, $E_\alpha(p_t(\boldsymbol{\xi})) = \mathrm{MMD}^\alpha_{\mu_t}[P, Q]$, where $p_t(\boldsymbol{\xi}) = \mu_t(\boldsymbol{\xi})/\mathrm{d}\boldsymbol{\xi}$.

Simulations on Synthetic Data-Set

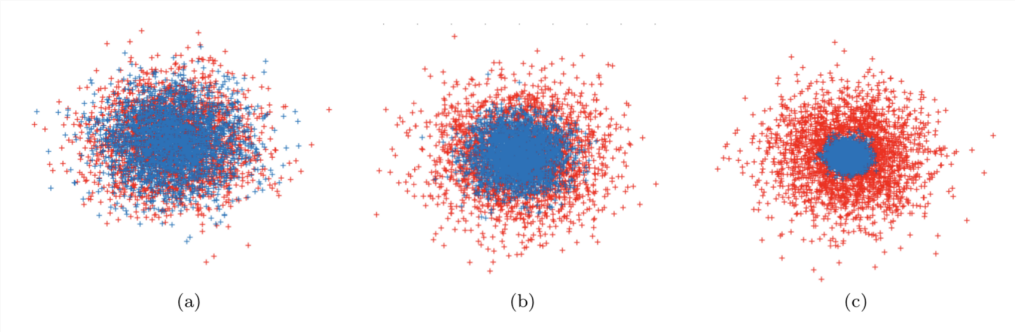


Figure 4: Panel (a): $\lambda = 0.1$, Panel (b): $\lambda = 0.5$, and Panel (c): $\lambda = 0.9$.

- Consider the problem of two sample test between two Gaussian distributions
  1. $P_0 = \mathsf{N}(\mathbf{0}, (1-\lambda)\boldsymbol{I}_{d\times d})$.
  2. $P_1 = \mathsf{N}(\mathbf{0}, (1+\lambda)\boldsymbol{I}_{d\times d})$.
- $\lambda \in [0, 1]$ controls the distance between these two distributions.
- Given samples $X_1, \cdots, X_m \sim_{\text{i.i.d.}} P_0$ and $Z_1, \cdots, Z_n \sim\sim_{\text{i.i.d.}} P_1$, we want to decide between the following hypotheses
  1. **Null Hypothesis** H_0: $P_0 = P_1$ $(\lambda = 0)$
  2. **Alternative Hypothesis** H_1: $P_0 \neq P_1$ $(\lambda > 0)$

Simulations on Synthetic Data-Set

- We design a test statistics as below

$$\mathcal{T}\left(\{\boldsymbol{V}_i\}_{i=1}^m, \{\boldsymbol{W}_i\}_{i=1}^n\right) \stackrel{\text{def}}{=} \begin{cases} \mathsf{H}_0 & \text{if } \widehat{\mathrm{MMD}}_K\left[\{\boldsymbol{V}_i\}_{i=1}^m, \{\boldsymbol{W}_i\}_{i=1}^n\right] \leq \tau \\ \mathsf{H}_1 & \text{if } \widehat{\mathrm{MMD}}_K\left[\{\boldsymbol{V}_i\}_{i=1}^m, \{\boldsymbol{W}_i\}_{i=1}^n\right] > \tau, \end{cases}.$$

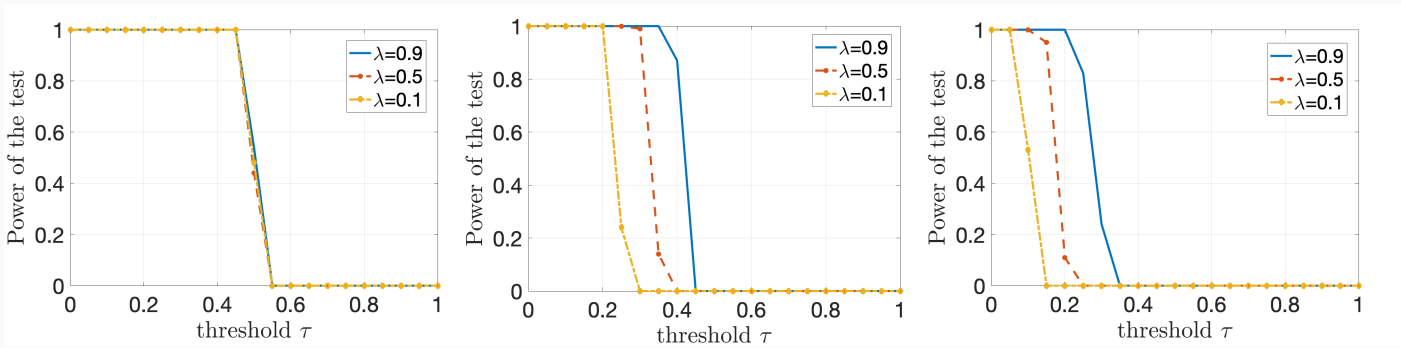


Figure 5: The statistical power, $\mathbb{P}(\text{reject } \mathsf{H}_0 | \mathsf{H}_1 \text{ is true})$, versus the threshold τ for the binary hypothesis testing via the unbiased estimator of the kernel MMD. Panel (a): Trained kernel using the two-phase procedure with the particle SGD and an auto-encoder, Panel (b): Trained kernel with an auto-encoder and a fixed Gaussian kernel with the bandwidth $\sigma = 1$, Panel (c): Untrained kernel without an auto-encoder.

Simulations on Real Data-Set: Qualitative Assessment

(a) (b) (c)

Figure 6: Sample generated images using CIFAR-10 and MNIST data-sets.Panel (a): Proposed MMD GAN with an *automatic* kernel selection via the particle SGD, Panel (b): MMD GAN with an auto-encoder optimization in conjunction with a mixed RBF Gaussian kernel, where the Gaussian bandwidths are *manually* tuned, Panel (c): MMD GAN with a single RBF Gaussian kernel with an auto-encoder optimization in conjunction with a single RBF Gaussian kernel where the Gaussian bandwidth is *manually* tuned.

Simulations on Real Data-Set: Quantitative Assessment

Method	FID (↓)	IS (↑)
MMD GAN (Gaussian)	67.244 ± 0.134	5.608 ± 0.051
MMD GAN (Mixture Gaussian)	67.129 ± 0.148	5.850 ± 0.055
Our Algorithm	$\mathbf{65.059 \pm 0.153}$	$\mathbf{5.97 \pm 0.046}$
Benchmark	-	11.237 ± 0.116

Table 1: Comparison of the quantitative performance measures of MMD GANs with different kernel learning approaches.

Conclusion

- Many machine learning tasks deal with the problem of learning a map between two domains.
- Machine learning systems divide into the supervised and unsupervised models based on the training samples. The hybrid version is often called semi-supervised model.
- Kernel methods provide an alternative method to deep learning to learn functions.
- However, there are model selection issues in kernel methods that need to be addressed. In this talk, we proposed a novel method to resolve those model selection issues.