Machine learning on the symmetric group

Jean-Philippe Vert

Google AI

MINES ParisTech
What if inputs are permutations?

• Permutation: a bijection
  \[ \sigma : [1, N] \rightarrow [1, N] \]

• \( \sigma(i) \) = rank of item \( i \)

• Composition
  \[ (\sigma_1 \sigma_2)(i) = \sigma_1(\sigma_2(i)) \]

• \( S_N \) the symmetric group

• \( |S_N| = N! \)
Examples

- Ranking data

- Ranks extracted from data

(histogram equalization, quantile normalization...)

[Image of wine bottles and people]
Examples

- Batch effects, calibration of experimental measures
Assume your data are permutations and you want to learn

\[ f : S_N \rightarrow \mathbb{R} \]

A solutions: embed \( S_N \) to a Euclidean (or Hilbert) space

\[ \Phi : S_N \rightarrow \mathbb{R}^p \]

and learn a linear function:

\[ f_\beta(\sigma) = \beta^\top \Phi(\sigma) \]

The corresponding kernel is

\[ K(\sigma_1, \sigma_2) = \Phi(\sigma_1)^\top \Phi(\sigma_2) \]
How to define the embedding $\Phi : S_N \rightarrow \mathbb{R}^p$?

- Should encode interesting features
- Should lead to efficient algorithms

- Should be invariant to renaming of the items, i.e., the kernel should be right-invariant

$$\forall \sigma_1, \sigma_2, \pi \in S_N, \quad K(\sigma_1 \pi, \sigma_2 \pi) = K(\sigma_1, \sigma_2)$$
Harmonic analysis on $\mathfrak{S}_N$

- A representation of $\mathfrak{S}_N$ is a matrix-valued function $\rho : \mathfrak{S}_N \rightarrow \mathbb{C}^{d_{\rho} \times d_{\rho}}$ such that
  \[ \forall \sigma_1, \sigma_2 \in \mathfrak{S}_N, \quad \rho(\sigma_1 \sigma_2) = \rho(\sigma_1) \rho(\sigma_2) \]
- A representation is irreductible (irrep) if it is not equivalent to the direct sum of two other representations
- $\mathfrak{S}_N$ has a finite number of irreps $\{\rho_\lambda : \lambda \in \Lambda\}$ where $\Lambda = \{\lambda \vdash N\}\footnote{\lambda \vdash N \iff \lambda = (\lambda_1, \ldots, \lambda_r) \text{ with } \lambda_1 \geq \ldots \geq \lambda_r \text{ and } \sum_{i=1}^r \lambda_i = N}$ is the set of partitions of $N$
- For any $f : \mathfrak{S}_N \rightarrow \mathbb{R}$, the Fourier transform of $f$ is
  \[ \forall \lambda \in \Lambda, \quad \hat{f}(\rho_\lambda) = \sum_{\sigma \in \mathfrak{S}_N} f(\sigma) \rho_\lambda(\sigma) \]
Bochner’s theorem

An embedding $\Phi : \mathbb{S}_N \to \mathbb{R}^p$ defines a right-invariant kernel $K(\sigma_1, \sigma_2) = \Phi(\sigma_1)^\top \Phi(\sigma_2)$ if and only there exists $\phi : \mathbb{S}_N \to \mathbb{R}$ such that

$$\forall \sigma_1, \sigma_2 \in \mathbb{S}_N, \quad K(\sigma_1, \sigma_2) = \phi(\sigma_2^{-1} \sigma_1)$$

and

$$\forall \lambda \in \Lambda, \quad \hat{\phi}(\rho_\lambda) \succeq 0$$
Some attempts

(Jiao and Vert, 2015, 2017, 2018; Le Morvan and Vert, 2017)
Let $\Phi(\sigma) = \Pi_{\sigma}$ the permutation representation (Serres, 1977):

$$[\Pi_{\sigma}]_{ij} = \begin{cases} 1 & \text{if } \sigma(j) = i, \\ 0 & \text{otherwise.} \end{cases}$$

Leads to new approaches for supervised quantile normalization (SUQUAN) and vector quantization.
SUQUAN = SUpervised QUANtile normalization

Suppose $\sigma = \text{rank}(x)$ with $x \in \mathbb{R}^N$

Rank-1 linear model on $\Pi_\sigma$:

$$f(\sigma) = \langle \Pi_\sigma, M \rangle_{\text{Frobenius}} \quad \text{with} \quad M = fw^\top$$

Then

$$f(\sigma) = \langle \Pi_\sigma, fw^\top \rangle_{\text{Frobenius}} = w^\top \Pi^\top_\sigma f$$

$\Pi^\top_\sigma f$ is the quantile normalization of $x$ with target quantile $f$

Learn $M$ amounts to learning both the linear model $w$ and the target quantile $f$
Example: CIFAR-10

- Discriminate images of horse vs. plane
- Different methods learn different quantile functions

![Images of horse vs. plane](image_url)
Limits of the SUQUAN embedding

- Linear model on $\Phi(\sigma) = \Pi_\sigma \in \mathbb{R}^{N \times N}$
- Captures first-order information of the form "i-th feature ranked at the j-th position"
- What about higher-order information such as "feature i larger than feature j"?
The Kendall embedding (Jiao and Vert, 2015, 2017)

\[ \Phi_{i,j}(\sigma) = \begin{cases} 
1 & \text{if } \sigma(i) < \sigma(j) , \\
0 & \text{otherwise.} 
\end{cases} \]
Geometry of the embedding

For any two permutations $\sigma, \sigma' \in S_N$:

- **Inner product**
  \[
  \Phi(\sigma)^T \Phi(\sigma') = \sum_{1 \leq i \neq j \leq n} 1_{\sigma(i) < \sigma(j)} 1_{\sigma'(i) < \sigma'(j)} = n_c(\sigma, \sigma')
  \]

  $n_c = \text{number of concordant pairs}$

- **Distance**
  \[
  \| \Phi(\sigma) - \Phi(\sigma') \|^2 = \sum_{1 \leq i, j \leq n} (1_{\sigma(i) < \sigma(j)} - 1_{\sigma'(i) < \sigma'(j)})^2 = 2n_d(\sigma, \sigma')
  \]

  $n_d = \text{number of discordant pairs}$
The Kendall kernel is

\[ K_T(\sigma, \sigma') = n_c(\sigma, \sigma') \]

The Mallows kernel is

\[ \forall \lambda \geq 0 \quad K_M^\lambda(\sigma, \sigma') = e^{-\lambda n_d(\sigma, \sigma')} \]

**Theorem (Jiao and Vert, 2015, 2017)**

The Kendall and Mallows kernels are positive definite right-invariant kernels and can be evaluated in \( O(N \log N) \) time.

*Kernel trick useful with few samples in large dimensions*
Kondor and Barbarosa (2010) proposed the diffusion kernel on the Cayley graph of the symmetric group generated by adjacent transpositions.

Computationally intensive ($O(N^{2N})$)

Mallows kernel is written as

$$K_M^\lambda(\sigma, \sigma') = e^{-\lambda n_d(\sigma, \sigma')} ,$$

where $n_d(\sigma, \sigma')$ is the shortest path distance on the Cayley graph.

It can be computed in $O(N \log N)$
Average performance on 10 microarray classification problems (Jiao and Vert, 2017).
Higher-order kernels (Jiao and Vert, 2018)

\[ \Phi(\sigma) = \Pi_{\sigma}^d \]

- For \( d = 1 \), this is the SUQUAN embedding
- For \( d = 2 \), this leads to a new weighted Kendall kernel, where weights can be optimized during training
Machine learning beyond vectors, strings and graphs
Different embeddings of the symmetric group
Scalability? Robustness to adversarial attacks? Differentiable embeddings?

MERCI!


The quantile normalization (QN) embedding

- Data: permutation $\sigma \in S_N$ where $\sigma(i) = \text{rank of item/feature } i$
- Fix a target quantile $q \in \mathbb{R}^N$
- Define $\Phi_q : S_N \to \mathbb{R}^N$ by

$$\forall \sigma \in S_N, \quad [\Phi_q(\sigma)]_i = q_{\sigma(i)}$$

- "Keep the order, change the values"
How to choose a "good" target distribution?

- Gaussian distribution (mean=0, sd=1)
- Uniform distribution
- Bigaussian distribution

Quantile functions:
- Gaussian
- Uniform
- Bigaussian
Learn after standard QN:

1. Fix \( q \) arbitrarily
2. QN all samples to get \( \Phi_q(\sigma_1), \ldots, \Phi_q(\sigma_n) \)
3. Learn a model on normalized data, e.g.:

\[
\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^N} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell_i (\beta^T \Phi_q(\sigma_i)) + \lambda \|\beta\|^2 \right\}
\]

Supervised QN (SUQUAN): jointly learn \( q \) and the model:

\[
(\hat{\beta}, \hat{q}) = \arg\min_{\beta, q \in \mathbb{R}^N} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell_i (\beta^T \Phi_q(\sigma_i)) + \lambda \|\beta\|^2 + \gamma \Omega(q) \right\}
\]
Learn after standard QN:

1. Fix $q$ arbitrarily
2. QN all samples to get $\Phi_q(\sigma_1), \ldots, \Phi_q(\sigma_n)$
3. Learn a model on normalized data, e.g.:

$$\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^N} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell_i \left( \beta^\top \Phi_q(\sigma_i) \right) + \lambda \|\beta\|^2 \right\}$$

Supervised QN (SUQUAN): jointly learn $q$ and the model:

$$\left( \hat{\beta}, \hat{q} \right) = \arg\min_{\beta, q \in \mathbb{R}^N} \left\{ \frac{1}{n} \sum_{i=1}^{n} \ell_i \left( \beta^\top \Phi_q(\sigma_i) \right) + \lambda \|\beta\|^2 + \gamma \Omega(q) \right\}$$
For $\sigma \in \mathbb{S}_N$ let the permutation representation (Serres, 1977):

$$[\Pi_\sigma]_{ij} = \begin{cases} 1 & \text{if } \sigma(j) = i, \\ 0 & \text{otherwise}. \end{cases}$$

Then

$$\Phi_q(\sigma) = \Pi_\sigma^T q$$
Linear SUQAN as rank-1 matrix regression

Linear SUQUAN therefore solves

\[
\min_{\beta, q \in \mathbb{R}^N} \left\{ \frac{1}{n} \ell_i \left( \beta^T \Phi_q(\sigma_i) \right) + \lambda \| \beta \|^2 + \gamma \Omega(q) \right\}
\]

\[
= \min_{\beta, q \in \mathbb{R}^N} \left\{ \frac{1}{n} \ell_i \left( \beta^T \Pi_{\sigma_i}^T q \right) + \lambda \| \beta \|^2 + \gamma \Omega(q) \right\}
\]

\[
= \min_{\beta, q \in \mathbb{R}^N} \left\{ \frac{1}{n} \ell_i \left( \langle q \beta^T, \Pi_{\sigma_i} \rangle > \text{Frobenius} \right) + \lambda \| \beta \|^2 + \gamma \Omega(q) \right\}
\]

A particular linear model to estimate a rank-1 matrix \( M = q \beta^T \)

Each sample \( \sigma \in \mathcal{S}_N \) is represented by the matrix \( \Pi_{\sigma} \in \mathbb{R}^{n \times n} \)

Non-convex

Alternative optimization of \( f \) and \( w \) is easy
Experiments: CIFAR-10

- Image classification into 10 classes (45 binary problems)
- $N = 5,000$ per class, $p = 1,024$ pixels

![Graph showing AUC on test set comparison]
Experiments: CIFAR-10

- Example: horse vs. plane
- Different methods learn different quantile functions

![Graphs and images showing original, median, SVD, and SUQUAN BND methods for horse vs. plane example.](image_url)
1. The Kendall embedding
Limits of the QN embedding

- Linear model on $\Phi(\sigma) = \Pi_\sigma \in \mathbb{R}^{N \times N}$
- Captures first-order information of the form "$i$-th feature ranked at the $j$-th position"
- What about higher-order information such as "feature $i$ larger than feature $j$"?
Another representation

$$\Phi_{i,j}(\sigma) = \begin{cases} 
1 & \text{if } \sigma(i) < \sigma(j), \\
0 & \text{otherwise.}
\end{cases}$$
The Kendall kernel is

\[ K_T(\sigma, \sigma') = \Phi(\sigma)^\top \Phi(\sigma') \]

The Mallows kernel is

\[ \forall \lambda \geq 0 \quad K_M^\lambda(\sigma, \sigma') = e^{-\lambda \|\Phi(\sigma) - \Phi(\sigma')\|^2} \]

**Theorem (Jiao and Vert, 2015, 2017)**

The Kendall and Mallows kernels are positive definite and can be evaluated in \( O(N \log N) \) time

*Kernel trick useful with few samples in large dimensions*
Proof

For any two permutations $\sigma, \sigma' \in S_N$:

- **Inner product**

$$\Phi(\sigma) \Phi(\sigma') = \sum_{1 \leq i \neq j \leq N} \mathbb{1}_{\sigma(i) < \sigma(j)} \mathbb{1}_{\sigma'(i) < \sigma'(j)} = n_c(\sigma, \sigma')$$

$n_c = \text{number of concordant pairs}$

- **Distance**

$$\| \Phi(\sigma) - \Phi(\sigma') \|^2 = \sum_{1 \leq i, j \leq N} \left( \mathbb{1}_{\sigma(i) < \sigma(j)} - \mathbb{1}_{\sigma'(i) < \sigma'(j)} \right)^2 = 2n_d(\sigma, \sigma')$$

$n_d = \text{number of discordant pairs}$

$n_c$ and $n_c$ can be computed in $O(N \log N)$ (Knight, 1966)
Kondor and Barbarosa (2010) proposed the diffusion kernel on the Cayley graph of the symmetric group generated by adjacent transpositions. Computationally intensive ($O(N^{2N})$)

Mallows kernel is written as

$$K_M^\lambda(\sigma, \sigma') = e^{-\lambda n_d(\sigma, \sigma')} ,$$

where $n_d(\sigma, \sigma')$ is the shortest path distance on the Cayley graph. It can be computed in $O(N \log N)$.
Average performance on 10 microarray classification problems (Jiao and Vert, 2017).
Constraints on $f$

- **Ridge**

$$
\mathcal{F}_0 = \left\{ f \in \mathbb{R}^p : \frac{1}{p} \sum_{i=1}^{p} f_i^2 \leq 1 \right\}.
$$

- **Non-decreasing**

$$
\mathcal{F}_{\text{BND}} = \mathcal{F}_0 \cap \mathcal{I}_0, \quad \text{where} \quad \mathcal{I}_0 = \left\{ f \in \mathbb{R}^p : f_1 \leq f_2 \leq \ldots \leq f_p \right\}
$$

- **Non-decreasing and smooth**

$$
\mathcal{F}_{\text{SPAV}} = \left\{ f \in \mathcal{I}_0 : \sum_{j=1}^{p-1} (f_{j+1} - f_j)^2 \leq 1 \right\}.
$$
SUQUAN-BND and SUQUAN-PAVA

We now focus on approximate algorithms to solve (8) in the case where \( F = FBND \) or \( F = FSPAV \). We then compare four methods to estimate \( w \) from \( n \) observations:

- SUQUAN-BND
- SUQUAN-PAVA
- Smoothed Pool Adjacent Violators algorithm (SPAV, Sysoev and Burdakov (2016)) as proximal operator.
- Accelerated proximal gradient optimization for alternate optimization in \( w \) and \( f \), monotonicity constraint on \( f \).

### Algorithm 2: SUQUAN-BND and SUQUAN-PAV

**Input:** \((x_1, y_1), \ldots, (x_n, y_n), f_{\text{init}} \in I_0, \lambda \in \mathbb{R}\)

**Output:** \( f \in I_0 \) target quantile

1. for \( i = 1 \) to \( n \) do
2. \( \text{rank}_i, \text{order}_i \leftarrow \text{sort} (x_i) \)
3. end for
4. \( w, b \leftarrow \arg\min_{w,b} \frac{1}{n} \sum_{i=1}^{n} \ell_i \left( w^\top f_{\text{init}}[\text{rank}_i] + b \right) + \lambda||w||^2 \) (standard linear model optimisation)
5. \( f \leftarrow \arg\min_{f \in F_{BND}} \frac{1}{n} \sum_{i=1}^{n} \ell_i \left( f^\top w[\text{order}_i] + b \right) \) (isotonic optimisation problem using PAVA as prox)
   OR
   \( f \leftarrow \arg\min_{f \in F_{SPAV}} \frac{1}{n} \sum_{i=1}^{n} \ell_i \left( f^\top w[\text{order}_i] + b \right) \) (smoothed isotonic optimisation problem using SPAV as prox)

Alternate optimization in \( w \) and \( f \), monotonicity constraint on \( f \)

Accelerated proximal gradient optimization for \( f \), using the Pool Adjacent Violators Algorithm (PAVA, Barlow et al. (1972)) or the Smoothed Pool Adjacent Violators algorithm (SPAV, Sysoev and Burdakov (2016)) as proximal operator.
A variant: SUQUAN-SVD

Algorithm 1: SUQUAN-SVD

Input:

\((x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^p \times \{-1, 1\}\)

Output: \(f \in \mathcal{F}_0\) target quantile

1. \(M_{LDA} \leftarrow 0 \in \mathbb{R}^{p \times p}\)
2. \(n_+ \leftarrow |\{i : y_i = +1\}|\)
3. \(n_- \leftarrow |\{i : y_i = -1\}|\)
4. for \(i = 1\) to \(n\) do
5. Compute \(\Pi_{x_i}\) (by sorting \(x_i\))
6. \(M_{LDA} \leftarrow M_{LDA} + \frac{y_i}{n_{y_i}} \Pi_{x_i}\)
7. end for
8. \((\sigma, w, f) \leftarrow \text{SVD}(M_{LDA}, 1)\)

- Ridge penalty (no monotonicity constraint), equivalent to rank-1 regression problem
- SVD finds the closest rank-1 matrix to the LDA solution:

\[
M_{LDA} = \frac{1}{n_+} \sum_{i : y_i = +1} \Pi_{x_i} - \frac{1}{n_-} \sum_{i : y_i = +1} \Pi_{x_i}
\]

- Complexity \(O(np \ln(p))\) (same as QN only)
Experiments: Simulations

- True distribution of $X$ entries is normal
- Corrupt data with a cauchy, exponential, uniform or bimodal gaussian distributions.
- $p = 1000$, $n$ varies, logistic regression.

![Graph showing AUC and Euclidean distance to original target quantile vs number of samples.](image)