Learning with Idealized Kernels

James Kwok Ivor Tsang

Department of Computer Science
Hong Kong University of Science and Technology
Hong Kong

Outline

- Introduction
- Proposed Method
- Experimental Results
- Conclusion

Kernel Choice

- Central role of the kernel
- Poor kernel choice can lead to significantly impaired performance
- Typically, selects a parametric kernel before learning
  - the associated kernel parameters can be learned
- Adapt also the form of the kernel itself
  - kernel matrix learning
    - semi-definite programming, alignment maximization, boosting
  - kernel function learning
    - hyperkernel

Kernel as Distance Metric

- Kernel defines an inner product (and, consequently, a distance metric) in the feature space $\mathcal{F}$

- Kernel design
  $\leftrightarrow$ finding a good distance metric
  $\leftrightarrow$ finding a set of good feature weights in $\mathcal{F}$

- Standard feature weighting methods
  - operate in the input space
  - number of parameters increases with input dimensionality
  - cannot be easily kernelized (dimensionality of $\mathcal{F}$ is usually very high)

Similar (or Dissimilar) Information

- Existing methods typically assume the availability of class label information in the training set
- However, this is sometimes difficult to obtain
- We may only know that certain pairs of patterns are similar (or dissimilar)
- Xing *et al.* (2003) proposed a distance metric learning method that utilizes such similarity information using convex programming
  - # parameters in Xing *et al.* scales linearly/quadratically with # features
  - computationally expensive when # features is large

Basic Idea

- Kernel defines the pairwise similarity between patterns
- Ideally, two patterns should be considered "similar" iff they belong to the same class $\longrightarrow$ ideal kernel [Cristianini 2002]

$$k^*(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} 1 & y(\mathbf{x}_i) = y(\mathbf{x}_j), \\ 0 & y(\mathbf{x}_i) \neq y(\mathbf{x}_j). \end{cases}$$

- As k^* is ideal, we can idealize a given kernel k by making it more similar to the ideal kernel
  - k^* can only be defined on the training patterns
  - Q: how to generalize this to patterns outside the training set?
- Learning a good kernel $\leftrightarrow$ distance metric learning
  - constrain the kernel adaptation to be a linear transform on $\mathcal{F}$

Idealizing the Kernel

- Idealized kernel :
$$\tilde{k} = k + \frac{\gamma}{2} k^*,$$
  - $\gamma \geq 0$ (to be determined)
  - as both k and k^* are valid kernels, so is $\tilde{k}$

- Assuming that $\gamma > 0$, then the alignment of $\tilde{k}$ will be greater than that of the original kernel if $\gamma > -\frac{\langle \mathbf{K}, \mathbf{K}^* \rangle}{n_+^2 + n_-^2}$ (n_+, n_- are $\#$ of $+$ve and -ve training examples)
  - if k is aligned in the "right" direction ($\langle \mathbf{K}, \mathbf{K}^* \rangle \geq 0$) or slightly "wrongly" ($\langle \mathbf{K}, \mathbf{K}^* \rangle$ is a small negative number), then the idealized kernel will have an increased alignment
  - extension to multi-class case is straightforward

Corresponding Distance Metric

- First consider k is the linear kernel
- Original inner product for $\mathbf{x}_i, \mathbf{x}_j \in \Re^p$: $k(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i' \mathbf{M} \mathbf{x}_j$
  - $\mathbf{M}$ is positive semi-definite
  - e.g., $\mathbf{M}$ is the identity matrix (Euclidean metric)
  - corresponding squared distance: $d_{ij}^2 = (\mathbf{x}_i - \mathbf{x}_j)' \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j)$
- Change k to $\tilde{k}$, new squared distance:

$$\tilde{\mathbf{K}}_{ii} + \tilde{\mathbf{K}}_{jj} - 2\tilde{\mathbf{K}}_{ij} = \begin{cases} d_{ij}^2 & y_i = y_j, \\ d_{ij}^2 + \gamma & y_i \neq y_j. \end{cases}$$

  - for patterns in different classes: distance increased

Changing the Distance Metric

- Modify the inner product to $\mathbf{x}_i'\mathbf{A}\mathbf{A}'\mathbf{x}_j$
  - $\mathbf{A}_{p\times p} = [\mathbf{a}_1, \ldots, \mathbf{a}_p]$, where $\mathbf{a}_i$'s are a set of "useful" directions
  - corresponding distance metric: $\tilde{d}_{ij}^2 = (\mathbf{x}_i - \mathbf{x}_j)'\mathbf{A}\mathbf{A}'(\mathbf{x}_i - \mathbf{x}_j)$
- Search for an $\mathbf{A}$ such that $\tilde{d}_{ij}^2$ approximates the desired distance metric obtained from the idealized kernel:

$$\tilde{d}_{ij}^2 \begin{cases} \leq d_{ij}^2 & y_i = y_j, \\ \geq d_{ij}^2 + \gamma & y_i \neq y_j. \end{cases}$$

  - same class: may get closer; different classes: pulled apart
- When only (dis)similarity information is available
  - $\mathcal{S}$: set containing similar pairs; $\mathcal{D}$: dissimilar pairs

$$\tilde{d}_{ij}^2 - d_{ij}^2 \begin{cases} \leq 0 & (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S}, \\ \geq \gamma & (\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{D}. \end{cases}$$

Primal

- May not be able to perfectly enforce this for all pairs in $\mathcal{D}$ and $\mathcal{S}$
  - introduce slack variables

- $\mathbf{A}$ performs projection onto a (hopefully small) set of useful features
  - small rank$(\mathbf{A})$ = rank$(\mathbf{A}\mathbf{A}')$ desirable
  - if $\mathbf{B} \equiv \mathbf{A}\mathbf{A}' = \mathbf{U}\boldsymbol{\Sigma}\mathbf{U}'$, then rank$(\mathbf{B})$ = rank$(\boldsymbol{\Sigma}) = \|\boldsymbol{\Sigma}\|_0$
  - approximate $\|\boldsymbol{\Sigma}\|_0$ by the Euclidean norm $\|\boldsymbol{\Sigma}\|_2 = \|\mathbf{B}\|_2$

$$\min_{\mathbf{B},\gamma,\xi_{ij}} \frac{1}{2}\|\mathbf{B}\|_2^2 + \frac{C_{\mathcal{S}}}{N_{\mathcal{S}}} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} \xi_{ij} + C_{\mathcal{D}}\left(-\nu\gamma + \frac{1}{N_{\mathcal{D}}} \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \xi_{ij}\right),$$

$$\text{subject to} \left\{ \begin{array}{ll} d_{ij}^2 \geq \tilde{d}_{ij}^2 - \xi_{ij}, & (\mathbf{x}_i,\mathbf{x}_j) \in \mathcal{S}, \\ \tilde{d}_{ij}^2 - d_{ij}^2 \geq \gamma - \xi_{ij}, & (\mathbf{x}_i,\mathbf{x}_j) \in \mathcal{D}, \\ \xi_{ij}, \gamma \geq 0. & \end{array} \right.$$

Dual

$$
\begin{aligned}
\max \quad & -\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} \alpha_{ij}(\mathbf{x}_i-\mathbf{x}_j)'\mathbf{M}(\mathbf{x}_i-\mathbf{x}_j) + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij}(\mathbf{x}_i-\mathbf{x}_j)'\mathbf{M}(\mathbf{x}_i-\mathbf{x}_j) \\
& -\frac{1}{2}\sum_{(\mathbf{x}_i,\mathbf{x}_j),(\mathbf{x}_k,\mathbf{x}_l)\in\mathcal{S}} \alpha_{ij}\alpha_{kl}((\mathbf{x}_i-\mathbf{x}_j)'(\mathbf{x}_k-\mathbf{x}_l))^2 \\
& -\frac{1}{2}\sum_{(\mathbf{x}_i,\mathbf{x}_j),(\mathbf{x}_k,\mathbf{x}_l)\in\mathcal{D}} \alpha_{ij}\alpha_{kl}((\mathbf{x}_i-\mathbf{x}_j)'(\mathbf{x}_k-\mathbf{x}_l))^2 \\
& +\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}}\sum_{(\mathbf{x}_k,\mathbf{x}_l)\in\mathcal{D}} \alpha_{ij}\alpha_{kl}((\mathbf{x}_i-\mathbf{x}_j)'(\mathbf{x}_k-\mathbf{x}_l))^2
\end{aligned}
$$

$$
\text{subject to} \left\{ \begin{array}{ll}
\frac{1}{C_\mathcal{D}}\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij} \geq \nu, & \\
0 \leq \alpha_{ij} \leq \frac{C_\mathcal{S}}{N_\mathcal{S}} & (\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}, \\
0 \leq \alpha_{ij} \leq \frac{C_\mathcal{D}}{N_\mathcal{D}} & (\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}.
\end{array}\right.
$$

- QP problem with $N_\mathcal{S}+N_\mathcal{D}$ variables (independent of $\mathbf{x}$'s dim)

"Support Vectors", "Error Pairs" and ν

- For $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{D}$,

$$\tilde{d}_{ij}^2 - d_{ij}^2 \begin{cases} = \gamma & 0 < \alpha_{ij} < \frac{C_\mathcal{D}}{N_\mathcal{D}}, \\ \geq \gamma & \alpha_{ij} = 0, \\ \leq \gamma & \alpha_{ij} = \frac{C_\mathcal{D}}{N_\mathcal{D}}. \end{cases}$$

  - α_{ij} nonzero but below upper bound: constraints exactly met
  - α_{ij} zero: constraints met with larger "margin", corresponding $(\mathbf{x}_i, \mathbf{x}_j)$ pair not used in solution (not "support vector")
  - α_{ij} at upper bound: constraints may be violated ("error")

- Similarly, for $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{S}$,

$$d_{ij}^2 - \tilde{d}_{ij}^2 \begin{cases} = 0 & 0 < \alpha_{ij} < \frac{C_\mathcal{S}}{N_\mathcal{S}}, \\ \geq 0 & \alpha_{ij} = 0, \\ \leq 0 & \alpha_{ij} = \frac{C_\mathcal{S}}{N_\mathcal{S}}. \end{cases}$$

- ν is a lower bound on the fraction of support vectors in $\mathcal{D}$ and an upper bound on the fraction of error pairs in $\mathcal{D}$

Heuristic for Computational Speedup

- Our QP problem has $N_{\mathcal{S}} + N_{\mathcal{D}}$ variables
- When similarity information is abundant
  - $N_{\mathcal{S}} + N_{\mathcal{D}}$ can be of $O(n^2)$ for a data set with n patterns
  - computationally expensive
- Simple heuristic inspired from locally linear embedding
  - for each pattern $\mathbf{x}$, its local neighborhood will be the most influential
  - only select the m closest $(\mathbf{x}, \mathbf{x}_j)$ pairs in $\mathcal{S}$ and $\mathcal{D}$ such that each of these $\mathbf{x}_j$'s is also within a radius of R from $\mathbf{x}$
  - $N_{\mathcal{S}} + N_{\mathcal{D}}$ will at most be of $O(n)$

Kernelize!

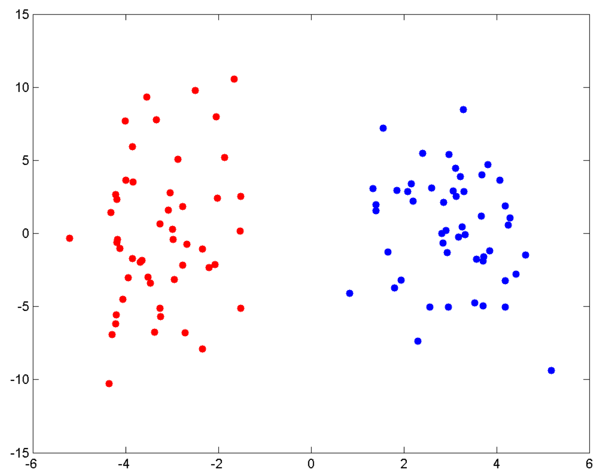
- Only inner products are required in our formulation
- Simply replace all the $\mathbf{x}$ by $\varphi(\mathbf{x})$ and then apply the kernel trick
- e.g.,
  - the idealized kernel $\tilde{\mathbf{K}}$ is then:

$$\begin{aligned}\tilde{\mathbf{K}}(\mathbf{x}_a, \mathbf{x}_b) &= -\sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{S}} \alpha_{ij}(K_{ai} - K_{aj})(K_{ib} - K_{jb}) \\ &\quad + \sum_{(\mathbf{x}_i,\mathbf{x}_j)\in\mathcal{D}} \alpha_{ij}(K_{ai} - K_{aj})(K_{ib} - K_{jb})\end{aligned}$$

  - similarly for the other expressions, such as the dual objective, distance metric, etc

Experiments

- Toy data set
  - one relevant feature, ten irrelevant features



- Real-world data sets (toy, colon, lymphoma, soybean, wine)
- Results based on averages over 50 random repetitions

With Class Label Information

- Use a subset for training, and the rest for testing
- $\mathcal{S}$ and $\mathcal{D}$: two patterns as similar if they belong to the same class; dissimilar otherwise.
- Classification using 1-nearest neighbor classifier
- Variations on different C_S, C_D settings $(\nu = 0.1)$

data set	$C_D \backslash \frac{C_S}{C_D}$	1	3	10	30	100
lymphoma (11.29%)	100	7.21%	7.53%	8.32%	6.84%	8.21%
	300	7.79%	8.53%	7.00%	7.58%	7.74%
	1000	6.42%	8.79%	8.58%	7.00%	6.32%
colon (26.82%)	100	16.92%	16.67%	17.67%	18.17%	18.17%
	300	18.17%	18.00%	17.42%	18.83%	15.92%
	1000	16.83%	19.00%	18.33%	18.92%	17.25%

data set	kernel	Euclidean metric	learned kernel	Xing *et al.*	tr align (before)	tr align (after)	tst align (before)	tst align (after)
toy	linear	28.50%	**3.08%**	3.33%	0.17	0.62	0.15	0.53
	rbf	27.75%	**7.92%**	-	0.70	0.77	0.69	0.73
colon	linear	29.83%	**14.67%**	-	0.15	0.28	0.31	0.33
	rbf	27.83%	**16.83%**	-	0.74	0.70	0.76	0.68
lym.	linear	14.17%	**8.11%**	-	0.19	0.30	0.18	0.20
	rbf	13.67%	**11.17%**	-	0.68	0.69	0.70	0.75
soybean	linear	2.82%	**0.12%**	0.59%	0.60	0.70	0.61	0.68
	rbf	2.82%	**1.17%**	-	0.77	0.86	0.79	0.84
wine	linear	28.03%	**10.13%**	22.58%	0.53	0.54	0.55	0.56
	rbf	27.62%	**26.82%**	-	0.71	0.58	0.66	0.50

- # parameters in Xing *et al.* scales quadratically with # features
  - cannot apply on colon, lymphoma nor RBF kernel
- Outperforms the original kernel and Xing *et al.*
- Training/test alignments typically improve after adaptation

With Similarity Information

- $\mathcal{S}$: random subset of all pairs of patterns belonging to the same class
- $\mathcal{D}$: same as in the previous experiment
- Classification using 1-nearest neighbor classifier

data set	kernel	Euclidean metric	learned kernel	Xing *et al.*
toy	linear	28.25%	**9.83%**	**9.83%**
	rbf	27.75%	**16.50%**	-
colon	linear	28.75%	**17.08%**	-
	rbf	27.83%	**22.67%**	-
lymphoma	linear	14.17%	**8.50%**	-
	rbf	11.00%	**9.94%**	-
soybean	linear	2.82%	**0.11%**	0.71%
	rbf	1.76%	**0.94%**	-
wine	linear	28.03%	**12.00%**	13.00%
	rbf	27.36%	**26.28%**	-

- Clustering using k-means clustering

$$\text{accuracy} = \sum_{i>j} \frac{1\{1\{c_i = c_j\} = 1\{\hat{c}_i = \hat{c}_j\}\}}{0.5n(n-1)} \quad \text{(Rand index)}$$

  – $1\{\cdot\}$ is the indicator function
  – n is the number of patterns
  – c_i: true cluster label for $\mathbf{x}_i$; $\hat{c}_i$: predicted label

data set	kernel	Euclidean metric	learned kernel	Xing *et al.*
toy	linear	55.33%	**100.00%**	98.11%
	rbf	50.16%	**100.00%**	-
colon	linear	77.15%	**82.28%**	-
	rbf	82.23%	**85.13%**	-
lymphoma	linear	79.50%	**88.86%**	-
	rbf	79.50%	**84.92%**	-
soybean	linear	83.63%	**100.00%**	**100.00%**
	rbf	84.45%	**100.00%**	-
wine	linear	71.87%	**77.63%**	73.46%
	rbf	72.04%	**73.12%**	-

Conclusion

- We propose idealizing a given kernel such that it becomes more similar to the ideal kernel
  - this is formulated as a distance metric learning problem that looks for a suitable linear transform (feature weighting)
- Requires only a training set with examples of similar and dissimilar pairs, but not explicit class label information
- Leads to a quadratic programming problem, and the number of variables is independent of the number of features
- Experimentally, improved performance on both classification and clustering tasks