Product Kernel Regularization Network

P. Kudová,T. Šámalová

Institute of Computer Science
Academy of Sciences of the Czech Republic

ICANNGA, 21st - 23rd March 2005,
Coimbra, Portugal

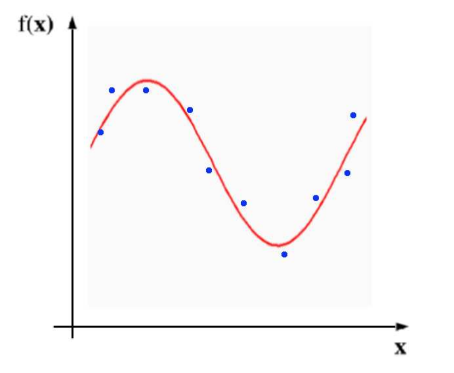
Outline

- Introduction
  - Learning from examples
- Theoretical background
  - Learning from data as minimization of functionals
  - Reproducing Kernel Hilbert Spaces
- Product Kernel Regularization Network
  - Motivation
  - Product kernel
  - Learning algorithm
  - Model selection
- Experimental results
  - Benchmark comparison
  - Flow rate prediction
- Conclusion

LEARNING FROM EXAMPLES

Problem statement

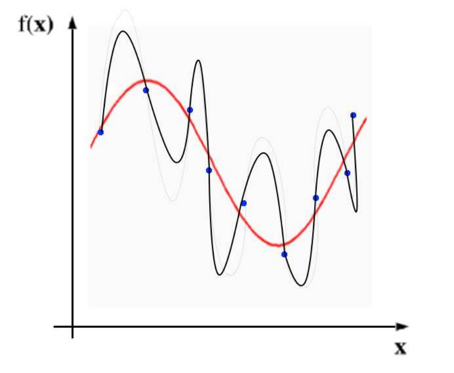
- **Given:** set of data samples $\{(\vec{x}_i, y_i) \in R^d \times R\}_{i=1}^N$
- **Our goal:** recover the unknown function or find the best estimate of it



LEARNING FROM EXAMPLES

Problem statement

- **Given:** set of data samples $\{(\vec{x}_i, y_i) \in R^d \times R\}_{i=1}^{N}$
- **Our goal:** recover the unknown function or find the best estimate of it



REGULARIZATION THEORY

Empirical Risk Minimization:

- find f that minimizes $H[f] = \frac{1}{N}\sum_{i=1}^{N}(f(\vec{x}_i) - y_i)^2$
- generally ill-posed
- choose one solution according to a priori knowledge (*smoothness, etc.*)

Regularization approach

- add a **stabiliser** $H[f] = \frac{1}{N}\sum_{i=1}^{N}(f(\vec{x}_i) - y_i)^2 + \gamma\Phi[f]$

REGULARIZATION THEORY

Empirical Risk Minimization:

- find f that minimizes $H[f] = \frac{1}{N}\sum_{i=1}^{N}(f(\vec{x}_i) - y_i)^2$
- generally ill-posed
- choose one solution according to a priori knowledge (*smoothness, etc.*)

Regularization approach

- add a **stabiliser** $H[f] = \frac{1}{N}\sum_{i=1}^{N}(f(\vec{x}_i) - y_i)^2 + \gamma\Phi[f]$

Reproducing Kernel Hilbert Space

Definition and properties

- RKHS is a Hilbert space of functions defined over $\Omega \subset \Re^d$ with the property that for each $x \in \Omega$ the evaluation functional on $\mathcal{H}$ given by $\mathcal{F}_x : f \rightarrow f(x)$ is bounded. *(Aronszajn, 1950)*

- This implies existence of positive definite symmetric function $K : \Omega \times \Omega \rightarrow \Re$ (*kernel function*) such that

$$\mathcal{H} = \mathcal{H}_K = \mathrm{comp}\{\sum_{i=1}^{n} a_i K_{x_i};\ x_i \in \Omega, a_i \in \Re\},$$

where comp means completion of the set.

Reproducing Kernel Hilbert Space

Application in learning

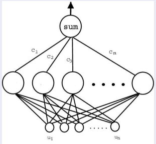
[Poggio, Smale, 2003]

- Data set: $\{(\vec{x}_i, y_i) \in R^d \times R\}_{i=1}^N$
- choose a symmetric, positive-definite kernel $K = K(\vec{x}_1, \vec{x}_2)$
- let $\mathcal{H}_K$ be the RKHS defined by K
- define the stabiliser by the norm $||\cdot||_K$ in $\mathcal{H}_K$

$$H[f] = \frac{1}{N}\sum_{i=1}^{N}(y_i - f(\vec{x}_i))^2 + \gamma||f||_K^2$$

- minimise $H[f]$ over $\mathcal{H}_K$ $\longrightarrow$ solution:

$$f(\vec{x}) = \sum_{i=1}^{N} c_i K_{\vec{x}_i}(\vec{x})$$



PRODUCT KERNEL REGULARIZATION NETWORK

Motivation

- different kernels are suitable for different data types
- data attributes are often of different types (temperature, color, age)

color	**temperature**	**age**
red	35.5	34
blue	36.0	56
red	36.9	45
...	...	...

PRODUCT KERNEL REGULARIZATION NETWORK

How to deal with attributes of different types?

- preprocessing
- convert everything to real values

Our approach

- divide the attributes to several subsets
- process the subsets separately
- select appropriate kernel for each subset
- allow difference not only in attribute type but also different properties – density, variance

PRODUCT KERNEL REGULARIZATION NETWORK

Product Kernels

[Aronszajn, 1950]

Let F_1 be an RKHS on Ω_1 with kernel K_1, F_2 an RKHS on Ω_2 with kernel K_2. Then

$$F = \mathrm{comp}\{\sum_{i=1}^{n} f_{1,i}(x_1) f_{2,i}(x_2)\}$$

is an RKHS on $\Omega_1 \times \Omega_2$ with kernel given by

$$K((x_1, x_2), (y_1, y_2)) = K_1(x_1, y_1) K_2(x_2, y_2),$$

where $x_1, y_1 \in \Omega_1$, $x_2, y_2 \in \Omega_2$.

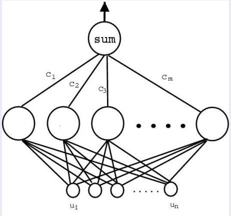
Completion by scalar product:

$$\langle \sum_{i=1}^{n} f_{1,i}(x_1) f_{2,i}(x_2), \sum_{j=1}^{m} g_{1,j}(x_1) g_{2,j}(x_2) \rangle = \sum_{i=1}^{n} \sum_{j=1}^{m} \langle f_{1,i}, g_{1,j} \rangle_1 \langle f_{2,i}, g_{2,j} \rangle_2$$

PRODUCT KERNEL REGULARIZATION NETWORK

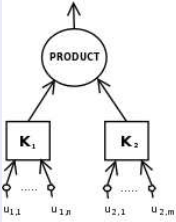
Network structure

- feedforward network with one hidden layer
- hidden layer of product units
- linear output layer



Product unit

- consists of several parts
- each part processes one group of attributes
- each part evaluates its own kernel function



PRODUCT KERNEL REGULARIZATION NETWORKS

Learning algorithm

Input: Data set $\{\vec{u_1}^i, \vec{u_2}^i, v^i\}_{i=1}^k \subseteq \Re^n \times \Re^m \times \Re$
Output: Product Kernel Regularization network.

1. Set the centers of kernels:

$$\forall i \in \{1, \ldots, k\} : \quad \begin{array}{l} \vec{c_1}^i \leftarrow \vec{u_1}^i \\ \vec{c_2}^i \leftarrow \vec{u_2}^i \end{array}$$

2. Compute the values of weigths $w_1, \ldots, w_k$:

$$(k\gamma I + K)\vec{w} = \vec{v},$$

   where I is the identity matrix,
   $K_{i,j} = K_1(\vec{c_1}^i, \vec{u_1}^j) \cdot K_2(\vec{c_2}^i, \vec{u_2}^j)$, and $\vec{v} = (v_1, \ldots, v_k)$, $\gamma > 0$.

MODEL SELECTION

Parameters of proposed algorithm

- kernel type
- kernel parameter(s) (i.e. width for Gaussian)
- regularization parameter γ

How we estimate these parameters?

- kernel type by user
- kernel parameter and regularization parameter by grid search and crossvalidation
- speed-up techniques: grid refining, lazy evaluation

EXPERIMENTS

Methodology

- two disjunct data sets for training and testing
- normalized error function

$$E = 100\frac{1}{N}\sum_{i=1}^{N} ||v^i - f(\vec{u_1}^{\,i}, \vec{u_2}^{\,i})||^2,$$

- LAPACK library was used for solving linear systems

Data Tasks

- benchmark – Proben1 data repository
- real life – prediction of flow rate of the river Ploučnice

EXPERIMENTS

Comparison with RN

	PKRN		**RN**	
	E_{train}	E_{test}	E_{train}	E_{test}
cancer1	2.739	**1.816**	2.658	1.875
cancer2	2.152	3.516	2.279	**3.199**
cancer3	2.374	**2.798**	2.348	2.873
glass1	6.141	8.590	4.899	**8.033**
glass2	5.269	**8.202**	4.570	8.317
glass3	3.691	**7.411**	4.837	7.691

Table: Error values for PKRN and RN on Proben1 data sets.

EXPERIMENTS

Prediction of flow rate

- prediction of the flow rate on the Ploučnice in North Bohemia, from origin (southwest part of the Ještěd hill) to the town Mimoň

- time series containing daily flow and rainfall values
- prediction of the current flow rate based on information from the previous one or two days
- 1000 training samples, 367 testing samples

EXPERIMENTS

Error values of PKRN

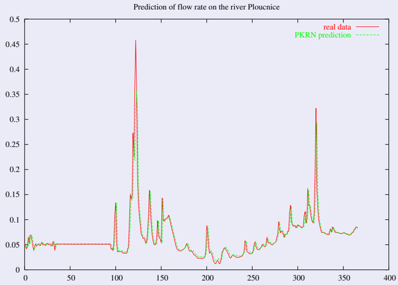
	pl1	pl2
E_{train}	0.057	0.109
E_{test}	0.048	0.097

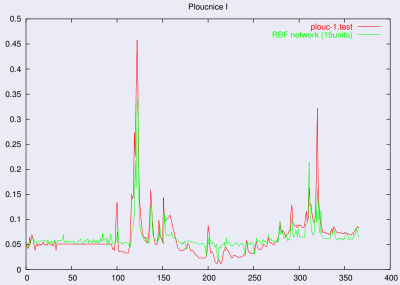
Comparison with conservative predictor

	PKRN	**CP**
E_{train}	0.057	0.093
E_{test}	0.048	0.054

EXPERIMENTS

Prediction of flow rate





CONCLUSION

Summary

- product of two kernel functions is a kernel function
- Product Kernel Regularization Network
- its behaviour demonstrated on experiments

Future work

- study properties and usability of other types of kernels
- automatic model selection (including type of kernel function)

Thank you for your attention.

Any questions?