Introduzione ai metodi kernel

Giorgio Valentini
DSI – Università degli Studi di Milano

1

Sommario

- Rappresentazione dei dati tramite kernel
- Caratteristiche dei kernel
- Kernel come misure di similarità
- Kernel e regolarizzazione
- Metodi di apprendimento automatico basati su kernel
- Combinazione di kernel

2

Rappresentazione dei dati

$$S = \{x_1, x_2, ..., x_n\}, S \subset U$$

Rappresentazione classica tramite features

$\phi(x) \to F$

$\phi(S) = \{\phi(x_1), \phi(x_2), ..., \phi(x_n)\}$

Es: se x è un gene:

$\phi(x) = ATGCTTAT...$

Rappresentazione tramite kernel

Dati rappresentati tramite confronti a coppie:

$k : U \times U \to R$

Data set S rappresentato tramite una matrice simmetrica:

$k_{i,j} = k(x_i, x_j)$

3

Vantaggi della rappresentazione dei dati tramite kernel:

- La rappresentazione dei dati come matrici quadrate non dipende dalla natura dei dati:
  - Lo stesso algoritmo può analizzare tipologie di oggetti diversi
  - Modularità fra la progettazione dei kernel e degli algoritmi per i kernel
  - Favorita l'integrazione dei dati (importante in bioinformatica!)
- La dimensionalità dei dati dipende solo dal numero degli oggetti
- La comparzione di oggetti a volte è più semplice di una loro esplicita rappresentazione (es: sequenze)

4

Caratteristiche dei kernel

- I metodi kernel operano su matrici semidefinite positive
- Un kernel semidefinito positivo (d'ora in poi semplicemente kernel) è una funzione simmetrica:

$$k(x_i, x_j) = k(x_j, x_i)$$

tale che per ogni n>0, $x_1,x_2,\ldots x_n$ in S e $c_1,c_2, \ldots c_n$ in R:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j k(x_i, x_j) = \mathbf{c}^T \mathbf{K}\mathbf{c} \geq 0$$

5

Kernel come prodotti interni

Kernel lineari:

$$\mathbf{x} = (x_1, x_2, ..., x_p) \qquad k_L(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}' = \sum_{i=1}^{p} x_i x_i'$$

k_L è un *kernel*: per ogni n>0, $\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots\boldsymbol{x}_n$ in R^p e $c_1,c_2, \ldots c_n$ in R:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j k_L(\mathbf{x}_i, \mathbf{x}_j) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j \mathbf{x}_i^T \mathbf{x}_j = \left\| \sum_{i=1}^{n} c_i \mathbf{x}_i \right\|^2 \geq 0$$

Se x non è un vettore?:

$$\phi(x) \rightarrow R^p \qquad k(x, x') = \phi(x)^T \phi(x')$$

k(x,x') è ancora un kernel?

6

Teorema (Aronszajn, 1950)

Ogni kernel può essere rappresentato tramite un prodotto interno in un opportuno spazio di Hilbert.

- Per ogni kernel esiste una mappa: $\phi(x) \to F$
  tale che $k(x, x') = <\phi(x), \phi(x')>$
- Non è necessario calcolare esplicitamente $\phi(x)$
  per calcolare il kernel

7

Prodotti interni e spazi di Hilbert (I)

Spazi vettoriali con prodotto interno:

uno spazio vettoriale X sui reali è uno spazio con prodotto interno se esiste una funzione simmetrica bilineare tale che:

$$\langle \mathbf{x}, \mathbf{x} \rangle \geq 0$$

Si può allora definire una norma su X: $\|\mathbf{x}\|_2 = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$

cui si può associare una distanza $d(\mathbf{x}, \mathbf{y}) = |\mathbf{x} - \mathbf{y}|_2$

Es: per lo spazio vettoriale R^p: $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{p} x_i y_i$

Uno spazio vettoriale X con una distanza è uno *spazio metrico*

8

Prodotti interni e spazi di Hilbert (II)

Spazio di Hilbert:

è uno spazio *F* con prodotto interno *completo* e *separabile*.

Completo: se ogni sequenza $\{h_n\}_{n>0}$ di Cauchy di elementi di *F* converge ad un elemento di *F*

Separabile: se per ogni $\varepsilon>0$ esiste un insieme finito $h_1, \ldots h_n$ in *F* tale che per tutti gli *h* in *F* vale:

$$\min_i \|h_i - h\| < \varepsilon$$

Le proprietà di completezza e separabilità assicurano che gli spazi di Hilbert sono isomorfi o a R^n (per qualche *n* finito) oppure allo spazio L_2 (dove n è infinito).

9

Prodotti interni e spazi di Hilbert (III)

Es. 1: X = R^n, **x** = $(x_1,x_2,\ldots,x_n)$, **y** = $(y_1,y_2,\ldots,y_n)$; c_i, $i=1,\ldots,n$ un inseme fissato di reali.

Un prodotto interno su X è il seguente:

$$<\mathbf{x},\mathbf{y}>=\sum_{i=1}^{n} c_i x_i\, y_i$$

Se $c_i=1$, $i=1,\ldots,n$, <**x**,**y**> è l'usuale prodotto interno (prodotto scalare) fra vettori in R^n

Es. 2: Lo spazio L_2:

X è lo spazio delle sequenze numerabili di numeri reali

x = $(x_1,x_2,\ldots,x_n,\ldots)$ t.c. $\sum_{i=1}^{\infty} x_i^2 < \infty$

Il prodotto interno fra 2 sequenze **x** e **y** è: $<\mathbf{x},\mathbf{y}>=\sum_{i=1}^{\infty} x_i\, y_i$

10

Prodotti interni e spazi di Hilbert (IV)

- **Es. 3**: Lo spazio $F = L_2(X)$ delle funzioni integrabili su un sottoinsieme compatto $X \subset R^n$:

$$L_2(X) = \left\{ f : \int_X f^2(x)dx < \infty \right\}$$

Per f,g appartenenti a $L_2(X)$ un prodotto interno è:

$$< f, g >= \int_X f(x)g(x)dx$$

11

Kernel come misure di similarità

- Se k(x,x') è "grande", allora x e x' sono "simili"
  Es: due sequenze sono simili quando esiste un "buon allineamento" fra loro.
  Es: due grafi sono simili quando condividono molti "cammini"
  Es: due vettori sono simili quando sono "allineati" (il coseno dell'angolo sotteso è 1)

12

La similarità è legata alla nozione di distanza

- Es: kernel gaussiano

$$k_G(\mathbf{x},\mathbf{y}) = \exp\left(-\frac{d(\mathbf{x},\mathbf{y})^2}{2\sigma^2}\right)$$

Essendo un kernel, per il Teo di Aronszajn esiste una ϕ tale che:

$$k_G(\mathbf{x},\mathbf{y}) = <\phi(\mathbf{x}),\phi(\mathbf{y})>$$

Per un kernel generale *k* si può dimostrare (provare…):

$$k(\mathbf{x},\mathbf{y}) = \frac{\|\phi(\mathbf{x})\|^2 + \|\phi(\mathbf{y})\|^2 - d(\phi(\mathbf{x}),\phi(\mathbf{y}))^2}{2}$$

Se tutti i $\phi(\mathbf{x})$ in $\phi(X)$ hanno la stessa norma, allora il kernel è una misura decrescente della distanza nello spazio delle feature $\phi(X)$

13

Kernel e spazi di Hilbert

- I kernel *k* sono associati a spazi funzionali H_k :

$$k \to H_k \subset \{f : X \to R\}$$

H_k sono spazi di Hilbert

- *Costruzione dello spazio funzionale* H_k dal kernel *k*:
  Dato un numero finito di punti $\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_n$, $\boldsymbol{x'}_1,\boldsymbol{x'}_2,\ldots,\boldsymbol{x'}_m$ e di pesi $\alpha_1,\alpha_2,\ldots\alpha_n$, $\alpha'_1,\alpha'_2,\ldots\alpha'_m$:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i,\mathbf{x}),\ g(\mathbf{x}) = \sum_{i=1}^{m} \alpha'_i k(\mathbf{x}'_i,\mathbf{x}), \qquad f, g \in H_k$$

$$< f, g >= \sum_{i=1}^{n}\sum_{j=1}^{m} \alpha_i \alpha'_j k(\mathbf{x}_i,\mathbf{x}'_j), \qquad \text{(prodotto interno)}$$

$$\|f\|^2_{H_k} = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j k(\mathbf{x}_i,\mathbf{x}_j), \qquad \text{(norma)}$$

14

Esempio: kernel lineari e spazio delle funzioni lineari

$$H_k = \left\{ f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} : \mathbf{w} \in R^p \right\}$$

Dato un numero finito di punti $\boldsymbol{x_1, x_2, \ldots, x_n}$

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} = \sum_{i=1}^{n} \alpha_i k_L(\mathbf{x}_i, \mathbf{x}) = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i{}^T \mathbf{x} \quad \Rightarrow \quad \mathbf{w} = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i$$

$$< f, g >= \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_i \alpha'_j k(\mathbf{x}_i, \mathbf{x}'_j) = \sum_{i=1}^{n} \sum_{j=1}^{m} \alpha_i \alpha'_j \mathbf{x}_i{}^T \mathbf{x}'_j \quad \text{(prod. interno)}$$

$$\|f\|^2_{H_k} = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \mathbf{x}_i{}^T \mathbf{x}_j = \mathbf{w}^T \mathbf{w}$$

$$\Downarrow$$

$$\|f\|_{H_k} = \|\mathbf{w}\| \quad \text{(norma)}$$

15

Reproducing Kernel Hilbert Space (RKHS)

Dalla costruzione precedente segue che *f* in un punto **x** può essere espressa come un prodotto interno nello spazio di Hilbert H_k: $f(\mathbf{x}) =< f, k(\mathbf{x},.) >$

Se $f(.)=k(\mathbf{x}',.)$ $\Rightarrow$ $k(\mathbf{x}, \mathbf{x}') =< k(\mathbf{x},.), k(\mathbf{x}',) >$

(proprietà di "riproduzione", da cui il nome RKHS.)

$\Downarrow$

H_k è un possibile spazio delle feature per il kernel *k* (si ricordi il Teo di Aronszajn).

16

Kernel, norme in H_k e regolarizzazione

Molti *metodi kernel* (comprese le *SVM*) possono essere interpretati come algoritmi che dato un insieme di oggetti *S*, risolvono il il seguente problema:

$$\min_{f \in H_k} L(f, S) + C\|f\|_{H_k}$$

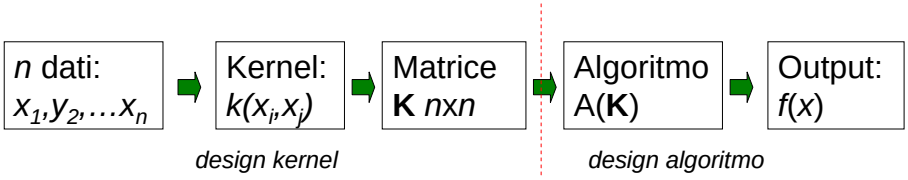
Dove *L* è una funzione di perdita associata al *rischio empirico* e *C* è un opportuno coefficiente di regolarizzazione.

$||f||_{Hk}$ è legata alla "*smoothness*" della funzione. La definizione di *smoothness* dipende dal kernel considerato.

17

Metodi kernel

- Algoritmi che hanno come input la matrice di similarità definita da un kernel:



- Modularità nella progettazione del kernel e dell'algoritmo

18

Il “trucco” del kernel

“Ogni algoritmo per dati vettoriali che può essere espresso in termini del prodotto interno fra vettori può essere implicitamente eseguito nello spazio delle feature associato al kernel, rimpiazzando i prodotti interni con valutazioni del kernel”

1. *Kernelizzazione* di metodi lineari (es: percettrone, linear discriminant analysis, PCA)
2. Applicazione di algoritmi definiti su vettori a *dati non vettoriali* (tramite kernel definiti su dati non vettoriali)

19

Esempio: calcolo della distanza fra 2 oggetti

Calcolo della distanza fra 2 sequenze o due molecole:

$d(\mathbf{x},\mathbf{y}) = \|\phi(\mathbf{x}) - \phi(\mathbf{y})\|$ (distanza di Hilbert fra le immagini)

Non è necessario calcolare ϕ:

$$\|\phi(\mathbf{x}) - \phi(\mathbf{y})\|^2 = <\phi(\mathbf{x}),\phi(\mathbf{x})> + <\phi(\mathbf{y}),\phi(\mathbf{y})> -2<\phi(\mathbf{x}),\phi(\mathbf{y})>$$

$$d(\mathbf{x},\mathbf{y}) = \sqrt{k(\mathbf{x},\mathbf{x}) + k(\mathbf{y},\mathbf{y}) - 2k(\mathbf{x},\mathbf{y})}$$

Es. 1: Come calcolare la distanza fra un oggetto ed un insieme di oggetti con il “trucco” del kernel?

Es. 2: Come calcolare la distanza fra due insiemi di oggetti con il “trucco” del kernel?

20

Teorema di rappresentazione

Il problema di ottimizzazione:

$$\min_{f \in H_k} \Psi(f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_n), \|f\|_{H_k})$$

ammette una rappresentazione della soluzione nella forma:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i k(\mathbf{x}_i, \mathbf{x})$$

- La soluzione giace in un sottospazio di H_k al più di dimensione n
- Il problema può essere riformulato come un problema di di ottimizzazione n-dimensionale: se $n<p$ (la dimensione dei pattern $\mathbf{x}$) si ha un vantaggio computazionale.
- Il termine di penalizzazione $||f||_{Hk}$ rende "smooth" la soluzione.

21

Tipologie di kernel

- *Kernel per vettori*:

Linear kernel: $k(x, x') = x^T x'$

Polynomial kernel: $k(x, x') = (x^T x' + c)^d$

Gaussian RBF kernel: $k(x, x') = \exp(-\|x - x'\|^2 / 2\sigma^2)$

sigmoid kernel: $k(x, x') = \tanh(\kappa x^T x' + \theta)$

- *Kernel per stringhe*: l'idea di fondo è di contare tutte le sottosequenze fino ad una certa lunghezza e costruire un vettore delle feature delle occorrenze.
- *Kernel che utilizzano modelli probabilistici parametrici* (HMM, grammatiche contex-free stocastiche) Ad es: Fisher Kernel.
- *Kernel per grafi*
- …

22

Operazioni sui kernel

- Una combinazione lineare di kernel è un kernel:

$$\lambda_1 k_1 + \lambda_2 k_2 = k \quad \Rightarrow \quad k \text{ è un kernel}$$

- Se una sequenza di kernel converge puntualmente ad una funzione *k* allora *k* è un kernel:

$$\forall x, x' \in X \quad \lim_{n\to\infty} k_n(x, x') = k(x, x')$$

- Una moltiplicazione punto a punto fra kernel è un kernel:

$$k_1(x, x')k_2(x, x') = k(x, x') \quad \Rightarrow \quad k \text{ è un kernel}$$

23

Combinazioni di kernel

•Sommare kernel è un semplice metodo per integrare dati eterogenei:

$$k = \sum_{i=1}^{c} k_i$$

Ogni k_i è definito su dati diversi.

•Una semplice generalizzazione è la somma pesata:

$$k = \sum_{i=1}^{c} \mu_i k_i$$

24

Costruzione di kernel da punteggi di similarità

Come costruire kernel da punteggi di similarità (che di per sé non sono kernel)?
Ad es: da misure di similarità fra sequenze o fra strutture 3D.

Un esempio: *Empirical kernel map* (Tsuda,1999):

1. Si scelgono $t_1, \ldots, t_p$ oggetti appartenenti a X (template)
2. Ogni oggetto x di X è rappresentato tramite un vettore di similarità:

$$x \in X, \phi(x) = (s(x,t_1),...s(x,t_p)) \in R^p$$

3. Il kernel è il prodotto interno fra vettori di similarità:

$$k(x,y) = \phi(x)^T \phi(y) = \sum_{i=1}^{p} s(x,t_i)s(y,t_i)$$

25

Support Vector Machine

- Metodo di apprendimento basato su kernel maggiormente noto ed applicato in molteplici campi della bioinformatica.
- Nel caso più semplice (kernel lineari per problemi di classificazione) risolve il problema di ottimizzazione:

$$\min_{f \in \{w^T x : w \in R^p\}} \left(\underbrace{C \max(0, 1 - yf(x))}_{\text{Hinge loss}} + \underbrace{\frac{1}{2}\|w\|^2}_{\text{regolarizzazione}} \right)$$

- Per il teorema di rappresentazione esiste una soluzione nella forma:

$$f(x) = \sum_{i=1}^{n} \alpha_i k(x_i, x) = \sum_{i=1}^{n} \alpha_i x_i^T x$$

26