UNIVERSITÀ DEGLI STUDI DI PADOVA
FACOLTÀ DI INGEGNERIA

Corso di Laurea Specialistica in Ingegneria Informatica

Ragionamento Qualitativo e Apprendimento Automatico per l'Analisi di Dati di Genomica

LAUREANDO: Matteo Zanini

RELATORE: prof. Silvana Badaloni

CORRELATORE: dott. Francesco Sambo

Anno Accademico 2010/2011

Gesù vide un uomo, seduto al banco delle imposte, chiamato Matteo, e gli disse: "Seguimi". Ed egli si alzò e lo seguì.
(Mt 9.9)

Introduzione

Con la pubblicazione della prima sequenza genomica, nel 1995, relativa al batterio *Haemophilus inuenzae*, si apre un'importante settore, l'ingegneria genetica, che, in questi ultimi anni, ha imperversato nell'ambito della ricerca scientifica. Un altro importante traguardo, nella storia di questa disciplina, è stato il completamento del progetto del genoma umano, nel 2003.

L'oggetto di studio di questa tesi, appartiene proprio a questa branca ed è la cosiddetta rete di regolazione genica. Questa è costituita da nodi, che corrispondono a dei geni e da lati, che segnalano un rapporto di regolazione tra i geni. Questo meccanismo di regolazione coinvolge diversi componenti che costituiscono la cellula (geni, proteine ed altre molecole) ed è un fenomeno molto complesso, del quale non si è ancora capito, in dettaglio, il funzionamento. Nelle ultime decadi, i *microarray* (misure simultanee dello stato di tutti i geni della cellula) si sono rivelati una rivoluzione in campo genomico, permettendo una raccolta enorme di dati e lo sviluppo di diversi algoritmi di analisi e per la ricostruzione delle reti di regolazione.

Scopo di questa tesi è la comparazione di due rappresentazioni dei profili di espressione genica, vale a dire la rappresentazione numerica e la rappresentazione simbolica. Per la loro valutazione, sono stati definiti due obiettivi distinti: il primo è cercare di ricostruire la rete di regolazione, a partire degli andamenti dei geni; il secondo è il reperimento di sottostrutture significative della rete, tramite procedimento di apprendimento automatico.

Questo lavoro può essere suddiviso in tre sezioni. La prima è una sorta d'introduzione, in cui sono descritte le caratteristiche dell'esperimento per l'estrazione dei profili genici (Capitolo 1); in seguito sono introdotte delle particolari sottostrutture significative, della rete di regolazione (Capitolo 2) e sono trattati dei metodi per la gestione dell'errore statistico, presente nei dati (Capitolo 3 e Capitolo 4).

La seconda sezione descrive le tecniche vere e proprie per l'elaborazione dei dati ottenuti, utilizzate per raggiungere i due obiettivi prefissati. Per prime sono state caratterizzate le due diverse rappresentazioni utilizzate; in seguito è sta-

ta descritta la tecnica utilizzata per la ricostruzione della rete di regolazione. Per l'altro obiettivo, invece, sono stati introdotte le SVM (*Support Vector Machine*), per l'apprendimento automatico nel riconoscimento delle sottostrutture FFL (*Feed Forward Loop*), particolari configurazioni di tre geni, che dal punto di vista biologico, rivestono un ruolo importante (Capitolo 5).
La terza ed ultima sezione riguarda i risultati ottenuti, applicando i metodi precedentemente descritti, su dataset reali, confrontando i metodi basati sulla rappresentazione numerica e quelli basati su quella simbolica (Capitolo 6 e Capitolo 7).
Come appendice al lavoro sono descritte alcune delle tecniche più utilizzate, per il processo di apprendimento automatico, nell'ambito dell'elaborazione dei dati di genomica (Appendice A).

Indice

Capitolo 1

La tecnologia dei DNA Microarray

1.1 Introduzione

La prima sequenza genomica ad essere stata pubblicata, nel 1995, fu quella del *Haemophilus influenzae*, un batterio gram-negativo, con un genoma di circa 1,8 milioni di basi. Successivamente, nel 1996, è stato completato il sequenziamento del primo genoma eucariotico, quello del lievito *Saccharomyces cerevisiae*, che comprende circa 13 milioni di basi, organizzate in sedici cromosomi. Infine, nel 2003, è stato completamente codificato il genoma umano, grazie all'omonimo progetto. Questi sono alcuni dei più importanti passi della ricerca, nel campo della biologia molecolare.

Nonostante i più sofisticati sistemi disponibili, allo stato attuale dell'arte, è possibile interpretare solo parzialmente gli elementi funzionali contenuti in un genoma e, ancor meno, si riesce a comprendere il significato dell'informazione genomica, nella sua globalità. Gli acidi nucleici offrono un metodo di indagine basato sulla specificità d'ibridazione di due eliche complementari, che fungono da sonde, per l'identificazione e la quantificazione di specifici mRNA. Il problema principale consiste nell'identificare le sequenze di DNA che sono trascritte in RNA messaggero (mRNA), per essere poi tradotte in proteine.

Il profilo trascrizionale riflette lo stato funzionale di una cellula, di conseguenza, capire in quali circostanze un gene si è espresso, è essenziale per comprenderne la funzione. Un aiuto importante è dato dalla regolazione dei geni, vale a dire quando si accendono e spengono, in risposta a particolari situazioni.

Lo studio dei profili di espressione genica, mediante tecnologia microarray, consente di quantificare contemporaneamente centinaia o migliaia di mRNA pre-

senti in un determinato campione, con un unico esperimento. Per comprendere meglio la trattazione di questa fase sperimentale, è opportuno prima fissare alcuni concetti base della biologia molecolare.
Le cellule sono le unità funzionali e strutturali biologiche di base. All'interno di esse si trova il citoplasma, una soluzione acquosa concentrata, suddivisa da un elaborato sistema di membrane e contenente enzimi, ioni e molecole disciolte, insieme ad un certo numero di organuli. Tra quest'ultimi, rivestono particolare interesse i ribosomi, che sono i siti in cui ha luogo l'assemblaggio e la sintesi proteica.
Nel citoplasma sono presenti anche i mitocondri, in cui avvengono le reazioni chimiche, che forniscono energia per le attività cellulari. La struttura più grossa ed importante, presente nella cellula, è il nucleo, che interagisce con il citoplasma, in modo da regolare le attività che si svolgono nella cellula. All'interno dell'involucro nucleare ha sede il nucleolo, il sito di formazione delle subunità ribosomiali, nonché della cromatina, sostanza formata da un complesso di proteine e di DNA.

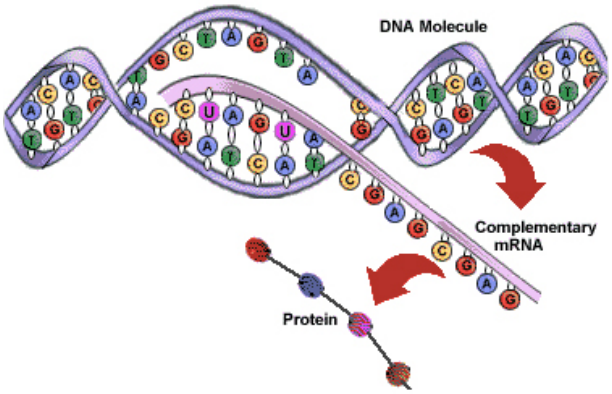


Figura 1.1: Processo di trascrizione di una porzione di DNA.

Il DNA (*Acido DesossiriboNucleico*) è una lunga molecola, costituita da due filamenti avvolti l'uno sull'altro e uniti da dei ponti infinitesimali, detti ponti idrogeno. I due filamenti sono costituiti da subunità ripetute di un gruppo fosfato e dello zucchero desossiribosio a cinque atomi di carbonio, mentre i ponti sono formati da una coppia di basi azotate. L'unione di uno zucchero desossiribosio, un gruppo fosfato e una base azotata, costituisce un nucleotide.
Esistono quattro tipi di basi: *adenina, timina, citosina* e *guanina* ed hanno la caratteristica di accoppiarsi sempre allo stesso modo, adenina con timina e citosina con guanina. Tutte e quattro insieme costituiscono una sorta di alfabeto, con il quale viene scandito il messaggio genetico: a seconda di come si presentano e si organizzano in triplette, si ha la formazione di un particolare *gene*, che è per l'appunto un segmento di DNA, in grado di trasmettere messaggi per la sintesi

delle proteine ed altre sequenze regolative. Una serie di tre nucleotidi (detta *codone*) codifica un amminoacido.
Il processo attraverso il quale il DNA viene tradotto in proteina consta di due fasi fondamentali: *trascrizione* e *traduzione*. Durante la prima fase, l'informazione viene *trascritta* ,da un filamento singolo di DNA in un filamento singolo di RNA, detto messaggero o mRNA. L'RNA messaggero è una molecola del tutto simile al DNA, solo che al posto della *timina* si trova un'altra base: l'*uracile*. Una volta trascritto, l'mRNA esce dal nucleo e si sposta nei ribosomi, dove ha luogo la sintesi proteica o *traduzione*.
I ribosomi sono costituiti da due subunità, formate da RNA ribosomiale e proteine (o rRNA). A questo punto interviene l'RNA di trasporto o tRNA, una molecola che assume la forma di un trifoglio e provvede al trasporto degli amminoacidi. Il tRNA è munito di una tripletta di basi, detta *anticodone*, specifica per l'amminoacido che trasporta. Durante la sintesi, il tRNA mette in corrispondenza ciascuna tripletta di basi (*codone*) del mRNA con il suo anticodone, in modo che ogni molecola di tRNA apporti l'amminoacido specifico, relativo al codone del mRNA a cui si attacca. In questo modo, in base alla sequenza dettata inizialmente dal DNA, le unità di amminoacidi, che vengono allineate una dopo l'altra, vanno ad assemblare la catena polipeptidica ossia la *proteina*.

1.2 Caratteristiche dei cDNA Microarray

Per comprendere la tecnica dei *microarray chip* è fondamentale specificare che, nella fase di trascrizione, ciascuna cellula produce RNA solamente per quei geni (ossia per quei segmenti di DNA) che sono attivi in quel momento; pertanto, un modo per indagare quali siano i geni attivi e quali quelli inattivi, in un determinato istante di tempo, sarà quello di analizzare l'RNA prodotto dalla cellula, ed è da questo punto che parte l'intuizione della *DNA microarray tecnology*.
L'idea base è misurare simultaneamente il livello di espressione di centinaia di geni, di una particolare popolazione di cellule o di un tessuto. Due sono i concetti chiave, alla base di questo processo di misura: la *retro trascrizione* e l'*ibridizzazione*. La retro trascrizione è il processo inverso rispetto alla trascrizione. Nella trascrizione, un filamento singolo di DNA viene usato come stampo, per la costruzione di una molecola di RNA. Durante la retro trascrizione, avviene il contrario: una molecola di mRNA viene *retro-trascritta* in DNA complementare o cDNA.
L'ibridizzazione è, invece, un processo di appaiamento delle due distinte eliche di DNA o RNA. Tale processo è basato sul principio di appaiamento delle basi, che consente l'unione, ossia l'ibridizzazione, solo di segmenti di DNA o RNA tra loro complementari. La *microarray tecnology* viene usata per misurare il livello relativo di espressione dei geni, in un particolare tessuto, ibridizzando il cDNA

retro trascritto dal mRNA di una cellula, con delle sequenze di cDNA sintetico, create in laboratorio e depositate in un *microchip* (o *array*). Il cDNA a contatto con il chip, viene chiamato *sonda* (*probes*) mentre il cDNA, derivante dalla retro trascrizione del mRNA cellulare, prende il nome di *target*.
Tre sono i principali approcci per la fabbricazione di microarray:

1. *cDNA microarray*: questo approccio consiste nel depositare, tramite un robot, una soluzione contenente le sonde di DNA, sulla superficie del supporto solido. Le sonde possono essere costituite da cDNA (DNA complementare ottenuto da una trascrizione del mRNA, che ha una lunghezza di 200-2400 basi) a singolo filamento oppure da oligonucleotidi (corte sequenze di nucleotidi, con lunghezza di 50-100 basi), chimicamente pre-sintetizzati. I *microarray* fabbricati con questo procedimento, nel caso in cui le sonde siano costituite da cDNA, sono chiamati *"cDNA microarray"*;

2. *Affymetrix*: si sintetizzano gli oligonucleotidi direttamente *(in situ)* sulla superficie del microarray, un'operazione che è eseguita principalmente con tecniche di tipo fotolitografico *(tipica di Affymetrix)* e di stampa a getto *(metodo sviluppato da Rosetta Inpharmatics e concesso in licenza ad Agilent Technologies)* ;

3. *Microarray di Agilent con oligonucleotidi sintetizzati in situ*: questi *microarray* sono realizzati sintetizzando oligonucleotidi direttamente sulla superficie di un vetrino, con un processo di stampa a getto.

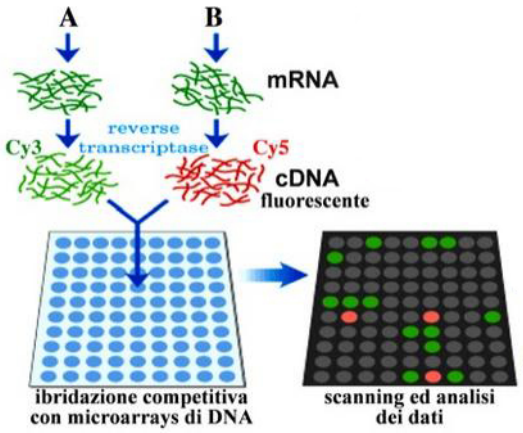


Figura 1.2: Processo di estrazione dei DNA microarray.

La *quantizzazione* o *quantificazione* è il processo che permette di ricavare, dall'insieme delle informazioni relative ad uno spot, un valore numerico, rappresentativo della concentrazione di mRNA, di un determinato gene, presente nel campione.

Dalla quantizzazione, si ricava il rapporto delle intensità assolute sui due canali, detto *fold change*, che serve ad avere informazioni sul livello di espressione di un canale rispetto all'altro. L'esito di questa operazione produce due immagini monocromatiche a 16 bit, corrispondenti a Cy3 e Cy5, memorizzate in un file TIFF, dove l'intensità di ogni pixel, su ogni canale, può essere quantificata tramite 65536 valori. Dalle immagini monocromatiche si ottiene l'immagine in falsi colori (rosso-verde) dei cDNA *microarray*.

1.3 Normalizzazione dei Dati

Molti fattori possono influire e distorcere i risultati di un esperimento di microarray:

1. Disomogeneità del processo di deposizione delle sonde;
2. Quantità iniziali diverse di RNA;
3. Diversa efficienza di incorporazione dei due fluorocromi, durante il procedimento di marcatura dei campioni;
4. Disomogeneità d'ibridizzazione sul vetrino;
5. Diversa efficienza di emissione dei due fluorocromi;
6. Diversa efficienza dello scanner, nel leggere i due canali di fluorescenza.

A questo punto è chiaro che è necessario effettuare un processo di normalizzazione dei dati, in modo da eliminare le distorsioni sistematiche (*bias*). Il processo di normalizzazione è necessario anche per confrontare dati provenienti da repliche dello stesso materiale. Solitamente in un esperimento di microarray le repliche fra vetrini possono essere di due tipi:

1. Repliche sperimentali: quando l'mRNA sui due vetrini proviene dalla stessa estrazione; si utilizzano per ottenere una stima migliore dell'espressione di un gene;
2. Repliche biologiche: quando l'mRNA proviene da campioni biologici dello stesso tipo ma distinti (ad esempio individui diversi) e consentono di stimare l'errore *random*.

È necessario che la normalizzazione tenga conto del disegno dell'esperimento: la sua scorretta applicazione, infatti, invalida completamente il dato e, di conseguenza, i risultati delle analisi sullo stesso. Si parla di normalizzazione *within-array* quando la tecnica scelta viene applicata, ad ogni vetrino, singolarmente, nell'intento di correggere gli errori sistematici su ogni array, preso come unità a sè

e indipendentemente dal disegno sperimentale, mentre si effettua una normalizzazione *between-arrays* quando si cerca di ottenere un dato uniforme, considerando sia il disegno sperimentale applicato che il tipo di campione biologico. In ciascuna di queste situazioni è necessario scegliere un gruppo di geni da utilizzare per la normalizzazione.

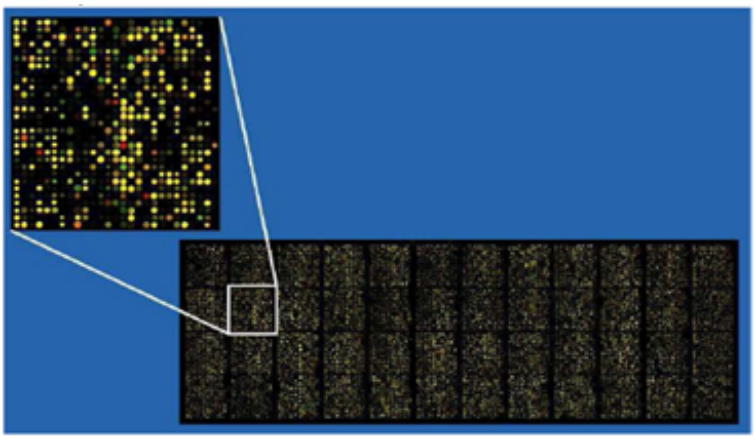


Figura 1.3: Cromartografia di un cDNA Microarray.

La struttura dei dati appena descritta mette in evidenza la peculiarità dell'analisi dei dati provenienti da esperimenti di *microarray*: siamo davanti ad un *dataset* con un numero molto elevato di geni, a fronte di una quantità decisamente ristretta di replicazioni a disposizione per ogni gene (paradigma del "*large p, small n*"). Proprio per il fatto che le replicazioni sono molto poche, se confrontate col numero di geni da analizzare, anche un piccolo errore sistematico può provocare distorsioni significative, nella fase d'inferenza.
La normalizzazione è un'operazione preliminare all'inferenza ed è necessaria per l'identificazione e la rimozione degli errori sistematici, all'interno dello stesso array e tra arrays distinti. Tali distorsioni possono essere introdotte in fase di preparazione del *microarray*, oppure durante l'esecuzione dell'esperimento, oppure ancora durante la scansione del risultato.
In letteratura, recentemente, sono stati proposti alcuni metodi, più o meno efficaci, per la normalizzazione di questi dati; possiamo classificare tali metodi in tre grandi categorie:

1. Metodi di normalizzazione globale: assumono che le intensità del canale rosso e del canale verde siano in relazione, attraverso un'opportuna costante. La normalizzazione quindi, consiste semplicemente nello spostamento della media (o mediana) della distribuzione dei logaritmi dei rapporti sullo zero

$$\log_2 \frac{x_g}{y_g} \to \log_2 \frac{x_g}{y_g} - m$$

2. Metodi di normalizzazione dipendenti dall'intensità (*intensity-dependent normalization*): la relazione tra i due canali non è più considerata costante, ma è una funzione che dipende dalla media dei logaritmi delle intensità, nei due canali:

$$log_2 \frac{x_g}{y_g} \rightarrow log_2 \frac{x_g}{y_g} - c\,(A)$$

dove

$$A = \frac{1}{2}\,(log_2 x_g + log_2 y_g)$$

La funzione $c(A)$ può essere, ad esempio, essere stimata tramite una regressione locale;

3. Metodi di normalizzazione spaziale (*spatial-dependent normalization*): la relazione tra i due canali è considerata dipendente dalle coordinate spaziali dello *spot* nell'*array*:

$$log_2 \frac{x_g}{y_g} \rightarrow log_2 \frac{x_g}{y_g} - c\,(X, Y)$$

Capitolo 2

Network Biologici

2.1 Introduzione

Come visto nel capitolo precedente, i dati relativi agli andamenti dei profili genetici sono soggetti ad errori stocastici. Questi sono di differente natura, ma possono essere raggruppati in due macroclassi:

1. Errori intrinseci al processo biologico, che sottostà alla produzione della proteine;
2. Errori dovuti al processo di estrazione dei dati.

I motivi e le problematiche, relativi a quest'ultima categoria, sono stati già discussi nel capitolo precedente. Per quanto riguarda, invece, il processo biologico, è necessario descrivere le caratteristiche che determinano il fattore aleatorio, presente nel processo di regolazione genetica.
La concentrazione di una proteina X, all'interno di una popolazione di cellule identiche, varia di cellula in cellula, a causa di processi stocastici. Per questa ragione la dinamica dei livelli di espressione delle proteine è soggetta a variazioni stocastiche.
Un'importante fonte di rumore è un disturbo di tipo estrinseco, per colpa del quale la capacità di produzione delle proteine e il sistema di regolazione dei geni subisce delle fluttuazioni col passare del tempo. In aggiunta del rumore estrinseco, ne esiste anche uno di tipo intrinseco, dovuto ad una variazione stocastica nella trascrizione e nella traslazione dei geni.
La distribuzione di una proteina in una cellula ha un andamento spesso simile alla distribuzione gaussiana, risultante dal prodotto di variabili stocastiche. Questo accade anche per la produzione delle proteine, che risulta dal prodotto dei processi di trascrizione e traslazione.

La tipologia dei collegamenti tra nodi contribuisce ulteriormente a questa variabilità. Nel caso di una struttura di geni, costituita da un ciclo con feedback negativo, le fluttuazioni vengono attenuate, mentre un auto regolazione positiva incrementa queste fluttuazioni.

La cellula è un sistema costituito da diverse migliaia di proteine, che interagiscono fra loro, ognuna della quali adempie a degli specifici compiti. Per rappresentare gli stati dell'ambiente interno, che costituisce una cellula, è possibile ricorrere ad una rappresentazione, basata sul comportamento di una serie di proteine, dette trascrittori, la cui funzione è molto importante nella dinamica della cellula.

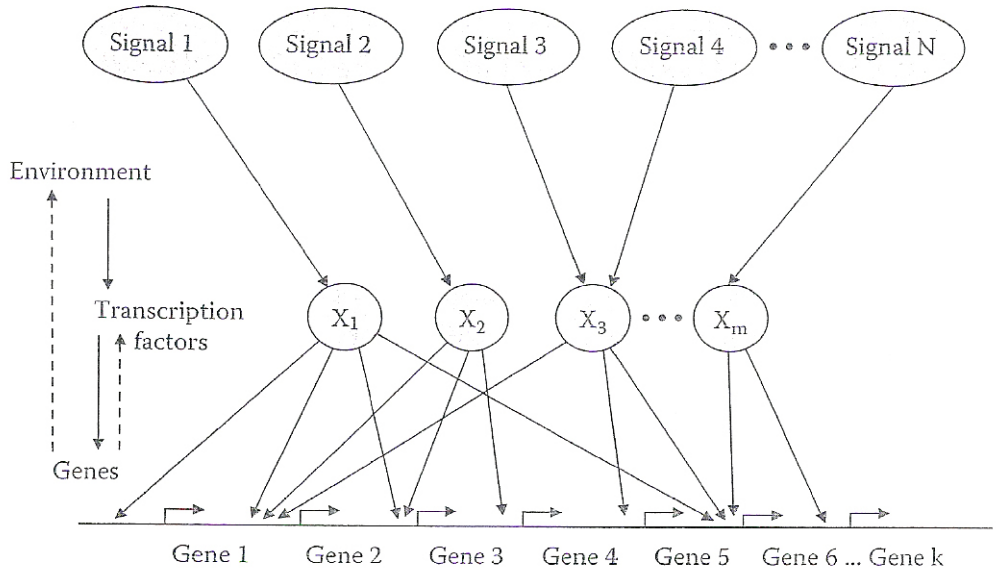
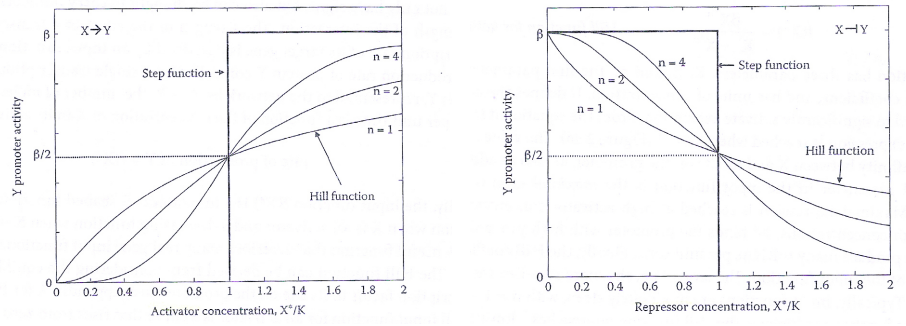


Figura 2.1: Rete di Regolazione Genica.

A seconda del tipo, in particolare del segno della regolazione, i fattori di trascrizione si dividono in promotori e repressori. Nel caso in cui il tasso di produzione, di un determinato gene, sia regolato in maniera positiva da un fattore di trascrizione, si parla di promotore, mentre nel caso contrario si parla di repressore.

A partire da questi concetti è possibile costituire una rete di trascrizione, costituita da nodi e archi orientati e dotati di un attributo. Nella corrispondenza con il sistema della cellula si ha che ogni nodo rappresenta un gene, i segnali d'ingresso portano con sé informazioni relative allo stato del sistema, mentre gli archi codificano le interazioni del sistema. Si ha che, usando la grammatica dei grafi, due geni X e Y sono correlati tra loro, $X \to Y$, che significa che la produzione del gene X è un fattore di trascrizione che influenza positivamente il grado di produzione del gene Y. L'attributo associato ad ogni arco è utilizzato per deter-



(a) (b)

Figura 2.2: Hill Function per regolazione positiva (attivatrice (a)) e negativa (repressiva (b)).

minare il segno, positivo o negativo, di questa regolazione, per poter distinguere tra promotori e repressori.

Analizzando la dinamica delle regolazioni tra geni ed in particole il tempo necessario, nei sistemi reali, a raggiungere lo stato d'equilibrio, si nota che si possono distinguere tra dinamiche molto lente e altre molto più veloci. Questa distinzione permette, tra l'altro, di suddividere le relazioni tra geni in due corrispettive classi. Mentre le relazioni più lente individuano le dipendenze sopracitate, che stanno alla base delle reti di trascrizione, le rimanenti, le più rapide, segnalano la presenza di ulteriori relazioni, che regolano altre attività interne alla cellula.

Un'altra particolarità, che emerge dall'analisi di sistemi reali, è la corrispondenza, all'interno delle rete di trascrizione, di concetti molto sfruttati in ambito ingegneristico. Nei sistemi reali, ad esempio, sono presenti particolari nodi, che posseggono un numero elevato di archi uscenti. Questa proprietà, nell'ambito dell'*Information Retrieval* (IR), prende il nome di hub. In particolare, nei casi reali analizzati (Bacteria E. Coli), si vede che esiste maggiore correlazione tra il segno degli archi uscenti da un nodo, rispetto a quelli entranti. Da questo segue che c'è una tendenza alla separazione tra le proprietà di attivazione e di repressione, in altre parole, si può dire che gli archi uscenti da uno stesso nodo tendono ad avere lo stesso segno di regolazione. Al contrario un nodo può essere correlato contemporaneamente da un fattore di trascrizione positivo e da uno negativo.

Per quanto riguarda i segnali d'ingresso, si parla di funzione gradino. Questa è una funzione monotona crescente o decrescente a seconda del segno della regolazione

$$f(X) = \frac{\beta X^n}{K^n + X^n} \; per\ un\ attivatore$$

$$f(X) = \frac{\beta}{1 + \left(\frac{X}{K}\right)^n} \; per\ un\ repressore$$

Questa può essere approssimata con dei gradini discontinui in cui il passaggio è istantaneo.
Prendiamo ad esame una semplice regolazione tra due singoli geni $X \rightarrow Y$ (si legge X regola positivamente Y).
In assenza di segnali d'ingresso X è inattivo e Y non è prodotto. Con l'introduzione di un segnale, X commuta rapidamente nella sua forma attiva, promuovendo la produzione del gene Y. A questo punto la produzione del gene Y avviene in un rapporto costante. Questa produzione è bilanciata da due processi, uno di degradazione e uno di diluzione, quest'ultimo dovuto alla crescita della cellula. Interrompendo improvvisamente il segnale d'ingresso, il processo di produzione del gene Y subisce un decadimento di ordine esponenziale, dal suo stato di equilibrio, precedentemente raggiunto tra i processi di degradazione e diluzione. Questo vale direttamente nel caso X sia un attivatore, ma la situazione in caso di un repressore si può facilmente ricavare per analogia.

2.2 Network Motifs

Analizzando diverse reti di trascrizione genetica è possibile domandarsi se esistano delle particolari configurazioni che, oltre ad essere ricorrenti, abbiano una certa importanza nella dinamica della cellula. Un metodo per la valutazione della ricorrenza di una sottorete, sono le cosiddette reti casuali. Queste sono costituite da una struttura corrispondente alla rete di trascrizione globale: si ha lo stesso numero di geni e di archi, ma quest'ultimi sono assegnati in maniera casuale a coppie di geni. Valutando la frequenza della sottorete d'interesse nella rete globale, rispetto alla frequenza nella rete casuale, si può ricavare se questa sia presente in maniera più o meno frequente. In pratica si ha che la frequenza nella rete casuale rappresenta la normale aspettativa di ricorrenze di un struttura e, se questa è superata in maniera consistente, si può dedurre che la sottorete ha una qualche rilevanza.
I concetti di base di questa tecnica sono stati introdotti da Erdos ed Renyi e sono stati poi ampiamente sviluppati da Uri Alon [Alo07], ed in pratica in una rete casuale costituita da N nodi e E archi si ha che ogni arco, nel grafo completo, ha la probabilità $p = E/N2$ di essere presente. Nel caso sia appurato che una certa sottorete occorre frequentemente, rispetto al modello casuale, si parla di **network motif**.
La sottostruttura più semplice da analizzare è quella costituita da un singolo nodo. Nel caso del *E. coli* si ha una certa frequenza di nodi autoregolati negativamente. La frequenza aspettata, tramite rete casuale, è pari a $p = 1/N$, per la singola autoregolazione. Nel caso invece di k occorrenze la probabilità è

$$p(k) = \left(\begin{array}{c} E \\ k \end{array} \right) \frac{1}{N} \left(1 - \frac{1}{N}\right)^{E-k}$$

Analizzando ora la frequenza reale, con quella ottenuta dalla precedente formula, si osserva che le occorrenze reali superano di molto quelle aspettate e che quindi, l'autoregolazione negativa può essere considerata un network motif. Resta ora da motivare questa ricorrenza, dal punto di vista biologico, infatti, nel caso in cui questa configurazione non apportasse nessun miglioramento nei processi di produzione del gene autoregolato, al momento della selezione sarebbe scartata, a favore di altre configurazioni più utili. Nel caso in questione, è possibile verificare che un cappio negativo porta ad un tempo di risposta del gene più ridotto. Per similarità si ha che con un'autoregolazione positiva il tempo di risposta aumenta. Un altro vantaggio dell'autoregolazione negativa è la robustezza, in quanto lo stato di equilibrio è meno sensibile alle fluttuazioni del segnale d'ingresso.

2.3 Feed Forward Loop

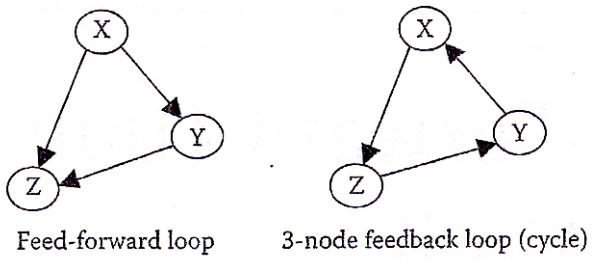
Passiamo ora ad analizzare tutte le possibili configurazioni costituite da tre geni. Esistono 13 possibili network motif. La prima considerazione da effettuare è che le reti reali tendono ad essere sparse, per cui risulta che il numero di archi è molto inferiore rispetto a quello di un grafo completo. Il motivo si riscontra analizzando il processo evolutivo della cellula, per cui i collegamenti superflui, che non portano miglioramenti alla stessa, tendono ad essere scartati al momento della selezione.

I Feed-Forward Loop risultano essere statisticamente le configurazioni più rilevanti, tra quelle costituite da tre nodi. Queste sottoreti appartengono ad una classe denominata triangolare. L'immagine 2.4 mostra tutte le possibili configurazioni di questa classe.

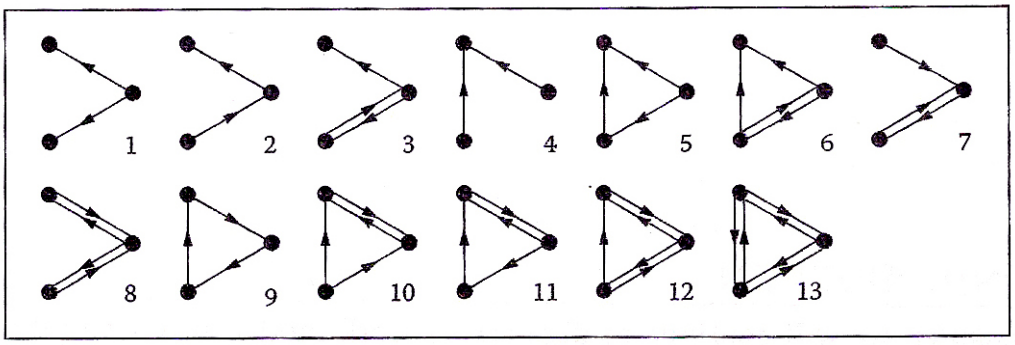
La suddivisione tra FFL coerenti e FFL incoerenti si ha analizzando il segno dell'arco tra X e Z: nel caso dei FFL coerenti, l'arco indiretto ha il segno uguale al prodotto degli altri due segni, mentre in quelle incoerenti ha il segno opposto. Questi concetti sono stati introdotti e sviluppati per la prima volta da [Alo07, Uri Alon].

Dal punto di vista statistico, la configurazione FFL più presente nelle reti reali è il primo tipo, sia per quanto riguarda le configurazioni coerenti che per quelle incoerenti, anche se in quest'ultimo caso la frequenza risulta inferiore.

Una particolare nota è necessaria per chiarire il comportamento del gene Z, nel FFL coerente del primo tipo. Sono principalmente due le possibili situazioni, in cui questo nodo possa venirsi a trovare: la sua attivazione può dipendere o dalla presenza contemporanea dei geni X e Y, oppure dalla presenza di almeno uno dei due. Questo vale sia per relazioni di tipo positivo, sia per quelle di tipo negativo. Il comportamento del gene Z, dovuto al collegamento parallelo dei geni X e Y, può essere sintetizzata come una porta logica, che unisce gli ingressi del nodo Z: nel caso in cui sia necessario la presenza contemporanea dei due ingressi, la porta



(a)



(b)

Figura 2.3: Insieme delle strutture possibili, costituite da tre nodi.

logica corrispondente sarà una AND, nel caso in cui, invece, sia necessario uno solo dei due ingressi la porta sarà un'OR.

Come nel caso dell'autoregolazione negativa, anche per il FFL vale l'associazione tra l'alta frequenza e l'apporto positivo, nel comportamento della cellula. Nel caso del FFL coerente del primo tipo, che usa una AND tra gli ingressi nel nodo Z, uno dei vantaggi che questa configurazione apporta, nel processo di trascrizione del gene Z, è la robustezza alle variazioni dei segnali d'ingresso. In altre parole, questa struttura è meno sensibile ad impulsi di durata eccessivamente breve, per i quali i geni X e Y non arrivano al loro stato attivo e, quindi, non influenzano il comportamento del gene Y.

Nel caso invece della presenza di una porta OR, il comportamento è simmetrico, in quanto il ritardo introdotto nel processo di trascrizione del gene Z si ha in corrispondenza dell'interruzione del segnale d'ingresso (con la porta AND il ritardo era in corrispondenza dell'attivazione dello stesso).

Per quanto riguarda, invece, la configurazione non coerente del primo tipo, il principale effetto positivo è una velocizzazione del tempo di risposta del sistema, in corrispondenza dell'attivazione del segnale d'ingresso. Come visto in precedenza, un altro metodo per ottenere questo tipo di vantaggio, è l'autoregolazione negativa, oppure anche un aumento del grado di degradazione nella cellula.

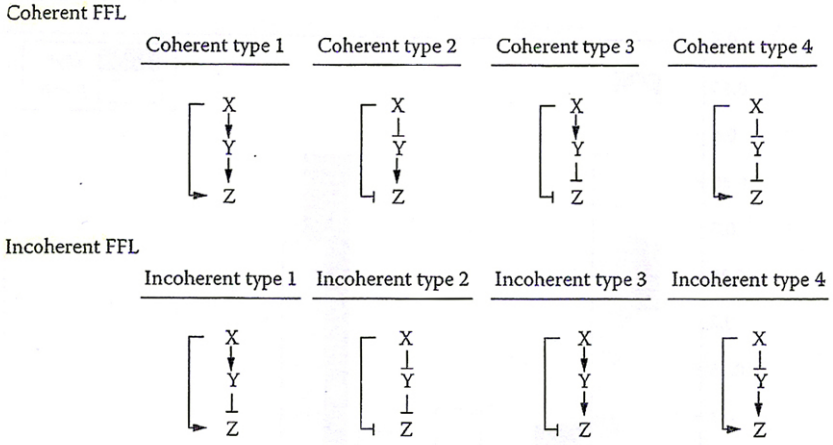


Figura 2.4: Tipologie di FFL. Come notazione si è utilizzata una freccia in corrispondenza di una regolazione positiva ed una barra nel caso di una regolazione negativa.

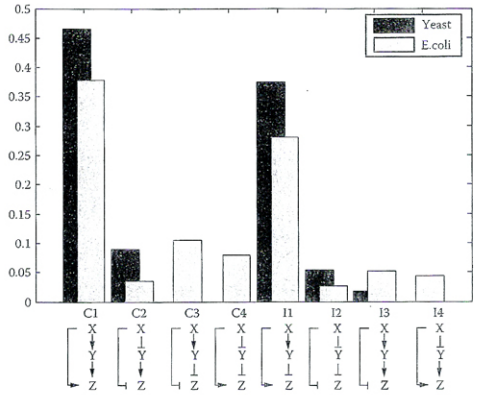


Figura 2.5: Frequenza delle tipologie di FFL in due dataset derivanti da esperimenti biologici.

Le reti di regolazione risultano, nei casi reali, di tipo sparso. Tipicamente solo lo 0,1% dei possibili collegamenti sono effettivamente presenti.

A questo punto è necessario introdurre alcuni concetti, per procedere con l'analisi delle proprietà grafiche delle reti di regolazione reali. Il grado di archi entranti in un nodo è il numero di nodi che puntano ad esso, il grado d'uscita è, invece, il numero di nodi puntati dallo stesso.

Fondendo questi concetti si può definire $\lambda = E/N$, come la connettività media della rete. Nella pratica si notano dei particolari geni, che controllano un gran numero di altri geni e che, per questo, vengono denominati hub. Un altro parametro di valutazione di una rete di trascrizione è la modularità, che rappresenta il grado

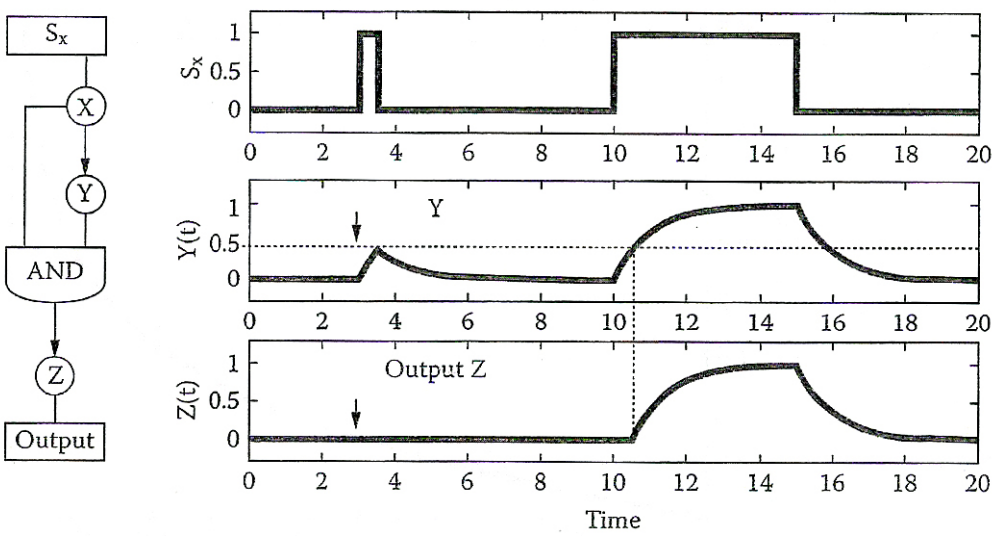


Figura 2.6: Struttura del FFL coerente del primo tipo.

di separabilità della rete in sottografi indipendenti.

Capitolo 3

Pre-elaborazione degli Andamenti Genetici

3.1 Introduzione

I dati, relativi ai profili genetici, utilizzati nella determinazione delle reti di trascrizione, si presentano come una collezione di serie numeriche. Come descritto nei capitoli precedenti, esistono diversi fattori che influiscono a perturbare i dati da elaborare e che quindi devono essere analizzati. In particolare, prima dell'eventuale determinazione delle relazioni tra geni, è necessarie effettuare una serie di tecniche di raffinamento, in modo da poter estrarre le componenti effettivamente significanti, dai dati soggetti al rumore.
In precedenza sono già state descritte delle operazioni eseguite a livello preliminare a tal fine, come ad esempio la ripetizione dell'esperimento d'estrazione dei dati, in modo che questi siano ottenuti come una media statistica, opportunamente pesata, in modo da minimizzare il contributo aleatorio.
Queste tecniche non sono comunque sufficienti ad eliminare completamente le componenti casuali presenti, per cui è necessario introdurre un processo di filtraggio dei dati. Prima di passare alla trattazione delle caratteristiche principali dei filtri utilizzati è necessario ribadire alcuni concetti derivanti dalla teoria dei sistemi.

Definizione 3.1 (filtro)**.** *Si definisce filtro un sistema che gode delle proprietà di linearità e di tempo invarianza.*

Dato un sistema $M : I \to O$, con I insieme di tutti i possibili segnali d'ingresso e O insieme di tutti i possibili segnali d'uscita, esso si dice lineare se sono soddisfatte le seguenti proprietà:

1. Additività: per ogni coppia di segnali d'ingresso x_1 e x_2 si ha che
   se vale $\begin{cases} M(x_1) = y_1 \\ M(x_2) = y_2 \end{cases}$ allora $M(x_1 + x_2) = y_1 + y_2$

2. Omogeneità: per ogni segnale d'ingresso x, preso un qualsiasi valore complesso α, si ha che se vale $M(x) = y$ allora $M(\alpha x) = \alpha y$

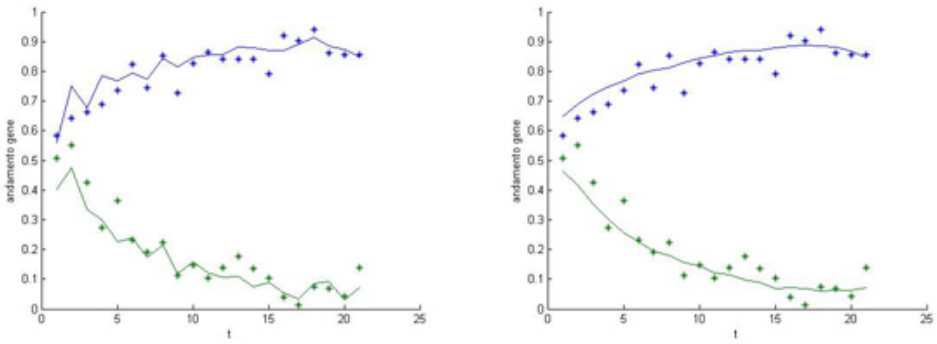
Un sistema si dice *tempo invariante* se una qualsiasi traslazione del segnale d'ingresso è riflessa tramite un identico shift del segnale d'uscita. Per cui, prendendo un segnale $x(t)$ tale che $M(x(t)) = y(t)$ e fissato un tempo di traslazione t_0, il sistema si dice tempo invariante se risulta

$$M(x(t - t_0)) = y(t - t_0)$$

In altre parole si può dire che il sistema reagisce sempre nel medesimo modo ad un segnale d'ingresso, indipendentemente dall'istante in cui questo segnale arriva al sistema.
Ritornando al concetto di filtro, il suo comportamento è sintetizzato dalla seguente equazione

$$y(t) = \int_{-\infty}^{+\infty} g(t - u)\, x(u) du$$

Da questa equazione si nota che la funzione g è quella che connota totalmente il comportamento del sistema, infatti, grazie a questa è sempre possibile ricostruire, dato un segnale d'ingresso, il corrispettivo segnale d'uscita, tramite un'operazione di convoluzione. Proprio in base alle proprietà e alle caratteristiche di questa funzione, si costituiscono diverse tipologie di filtri. Per il nostro caso in esame siamo in una situazione in cui il segnale che si vuole analizzare ha una componente significativa, quella che si vuole effettivamente analizzare, che varia in maniera lenta, se confrontata con la componente stocastica presente, la quale ha una dinamica molto più veloce.
Per descrivere questa differenza di velocità di transizione, risulta più chiaro ragionare nel campo delle frequenze, tramite trasformate di Fourier dei segnali. Dalla teoria scientifica, che analizza i processi biologici, che regolano le relazioni tra geni, ci si aspetta una banda limitata, mentre, per quanto riguarda il disturbo, la banda da considerare è infinita.
La risposta in frequenza dei filtri passa basso, svolge questo compito, eliminando il più possibile le componenti del segnale fuori della banda base, dove si trova l'effettiva porzione di segnale significativa.
La scelta del filtro da utilizzare è sempre soggetta alla valutazione del tipo di utilizzo che si richiede. In particolare si ha che l'uso più comune dei filtri passa basso è quello di ottenere le cosiddette funzioni di smoothing, vale a dire un'approssimazione del segnale, che elimini eventuali componenti aleatorie. Le condizioni



(a) (b)

Figura 3.1: Esempio di applicazione di un filtro ellittico (figura (a)) e di un filtro Savitzky-Golay (figura (b)) ad una distribuzione reale di dati.

iniziali in cui si trovano i segnali da analizzare, per questi motivi, sono riassumibili con lo presenza contemporanea di due componenti, una che rappresenta il segnale significativo, che generalmente varia lentamente, mentre la seconda componente rappresenta il rumore presente, la cui velocità di variazione è nettamente superiore a quella del segnale principale.

L'idea di base è quella di sostituire ogni punto della serie del segnale, con una media dei punti appartenenti ad un determinato intorno dello stesso. In questo modo si sfrutta la scarsa varianza del segnale principale, per ridurre tramite l'operazione di media, l'apporto del rumore. L'effetto dello smussamento deve sempre essere analizzato nell'ottica dell'uso che si ha dei dati, in quanto potrebbe succedere che, a seconda delle situazioni, risulti più conveniente utilizzare dati grezzi, piuttosto che dati elaborati, per non correre il rischio, ad esempio, di perdere dell'informazione nel passaggio intermedio.

I dataset esaminati in questa tesi, che saranno descritti maggiormente al momento del loro utilizzo, sono dataset sia reali, sia simulati, ma in entrambi i casi l'analisi della componente rumorosa è sempre importante. Per questa ragione risulta plausibile un raffronto tra le diverse tipologie di filtri. Si può notare inoltre, che questi passi di analisi, finora descritti, sono la trasposizione procedurale diretta dell'operato di un agente umano.

3.2 Filtro Savitzky-Golay

Il primo filtro che sarà analizzato è il filtro Savitzky-Golay. Questa tipologia deriva direttamente da una particolare formulazione del problema di smoothing nel dominio del tempo. Nella pratica, il filtro Savitzky-Golay è utilizzato per rendere visibile sia la larghezza, sia l'altezza delle linee spettrali di dati rumorosi.

Il filtro è applicato ad una serie di campioni equispaziati $f_i \equiv f(t_i)$ per cui vale $t_i \equiv t_0 + i\triangle$, con $\triangle$ costante spaziale e con $i = \ \ldots, -2,\ -1,\ 0,\ 1,\ 2, \ldots$.
Il più semplice filtro digitale possibile, che sia non ricorsivo e con risposta all'impulso finita, rimpiazza ogni campione f_i con una combinazione lineare g_i di un sottoinsieme di campioni vicini.

$$g_i = \sum_{n=n_L}^{n_R} c_n f_{i+n}$$

Si ha che n_L e n_R sono rispettivamente il numero di campioni di sinistra e di destra, da utilizzare per la combinazione lineare. Il filtro si dice causale se risulta $n_R = 0$, per cui la sommatoria non riguarda campioni successivi a quello preso in esame.
Per iniziare a comprendere il filtro Savitzky-Golay si può considerare la più semplice procedura per il calcolo della media. Fissato $n_L = n_R$ si calcola ogni g_i come l'effettiva media dei campioni tra f_{i-n_L} e f_{i+n_R}.
Questa procedura prende il nome di *moving window averaging* e corrisponde all'equazione precedente, posto $c_n = \frac{1}{n_L+n_R+1}$. Nel caso in cui la funzione principale del segnale sia costante. oppure monotona crescente o decrescente, nessuna distorsione viene aggiunta ai risultati. Una distorsione, comunque è aggiunta se la funzione dominante da analizzare ha una derivata seconda nulla. In corrispondenza di un massimo locale, ad esempio, la media a finestra mobile riduce sempre il valore risultante della funzione. Nel campo spettrografico questa riduzione corrisponde ad una diminuzione della larghezza e dell'altezza di una linea dello spettro. Dato che questo effetto riduce dei parametri caratteristici del segnale, bisogna cercare di evitare questo tipo di distorsione.
L'obiettivo del filtro Savitzky-Golay è proprio quello di eliminare questa manomissione. In pratica si sostituisce alla finestra di tipo rettangolare, in cui la pesatura dei suoi coefficienti è costante, con una finestra di tipo polinomiale o di ordine superiore, ad esempio quadratica oppure del quarto ordine. Per ogni campione f_i si effettua un fitting polinomiale ai minimi quadrati dei $n_L + n_R + 1$ campioni della finestra scorrevole e si prende come coefficiente g_i l'elemento alla posizione i-esima. Il resto dei punti, del polinomio calcolato, viene ignorato. Nel passaggio ad un campione successivo della successione, si effettua un nuovo calcolo completo di fitting ai minimi quadrati, ottenuto dalla trasposizione della finestra.
Dal punto di vista computazionale questo procedimento risulterebbe molto dispendioso, in quanto richiederebbe per ogni campione un numero di operazioni decisamente elevato. Per fortuna si può notare che il processo di fitting richiede solo l'inversione di una matrice. È possibile quindi effettuare a priori tutti i fitting, in quanto ognuno di questi è ottenibile come combinazione lineare di semplici fitting, in cui la matrice è composta di tutti zeri ed un unico valore uguale ad uno.

Questa è l'idea chiave attorno alla quale si sviluppa la costruzione del filtro. In particolare esistono dei set specifici di c_n, per cui risulta che l'equazione base del filtro soddisfa automaticamente il processo di approssimazione ai minimi quadrati, all'interno della finestra scorrevole.
Per derivare tali coefficienti basta pensare a come può essere ottenuto g_0: vogliamo approssimare un polinomio di grado M nel valore i, denominiamo con $a_0 + a_1 i + \cdots + a_M i^M$ i valori $f_{-n_L}, \ldots, f_{n_R}$. Allora g_0 sarà il valore del polinomio in corrispondenza di $i = 0$, a cui diamo il nome a_0.
La matrice relativa a questo problema diventa
$A_{i,j} = i^j \; i = -n_L \ldots n_R \; j = 0 \ldots M$
e le equazioni normalizzate per il vettore degli a_j, espresso in termini del vettore dei f_i, utilizzando la notazione matriciale, risultano
$\left(A^T \cdot A\right) \cdot a = A^T \cdot f$ oppure $a = \left(A^T \cdot A\right)^{-1} \cdot A^T \cdot f$
Possono essere espresse anche nella forma

$$\left\{A^T \cdot A\right\}_{i,j} = \sum_{k=n_L}^{n_R} A_{k,i} A_{k,j} = \sum_{k=n_L}^{n_R} k^{i+j}$$

e

$$\left\{A^T \cdot f\right\}_j = \sum_{k=n_L}^{n_R} A_{k,j} f_k = \sum_{k=n_L}^{n_R} k^j f_k$$

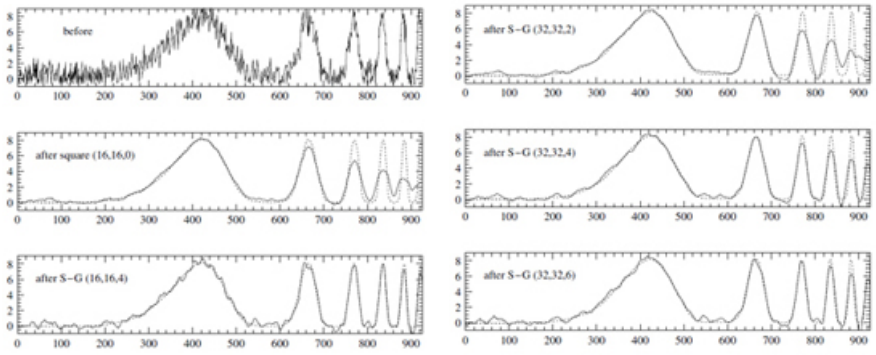
Dato che i coefficienti c_n sono i componenti a_0 in cui f è sostituita dal vettore unità e_n, $-n_L \leq n \leq n_R$ si ottiene

$$c_n = \left\{\left(A^T \cdot A\right)^{-1} \cdot \left(A^T \cdot e_n\right)\right\}_0 = \sum_{m=0}^{M} \left\{\left(A^T \cdot A\right)^{-1}\right\}_{0,m} \cdot n^m$$

Quest'equazione dimostra che è necessaria un'unica riga della matrice inversa, la quale può a sua volta essere ottenuta con un'unica riduzione, mediante scomposizione LU.
Il filtro Savitzky-Golay è contraddistinto da tre parametri, n_L, n_R e M, che determinano completamente il suo comportamento.
Una proprietà, che deriva direttamente dalla struttura propria del filtro Savitzky-Golay, è quella che lega il livello dello smoothing con la grandezza dei parametri n_L e n_R: più grandi sono questi parametri, maggiore sarà lo smoothig, in quanto per ogni campione risulta maggiore l'effetto dell'operazione di media e quindi di riduzione della componente stocastica.

3.3 Filtro Butterworth

Nel design di un filtro digitale, particolare importanza risiede nella scelta nell'ordine del filtro, in pratica nel numero di zeri e poli della funzione di trasferimento.



(a) (b)

Figura 3.2: Esempio di utilizzo di un filtro Savitzky-Golay su dati simulati.

Nella configurazione passa basso, si ha che l'ordine influenza direttamente lo scostamento della risposta reale all'impulso del filtro, con quella ideale. La transizione non è infatti immediata, come accade nella funzione reale, bensì preservando la continuità, decade più o meno velocemente, proprio a seconda dell'ordine del filtro. In un semplice filtro del primo ordine, ad esempio, la transizione avviene con una pendenza negativa di 20dB/decade. Se si aumenta il grado del filtro, questa pendenza aumenta, riducendo così quell'insieme di frequenze che, nella configurazione passa basso, pur essendo fortemente attenuate, stando alla caratteristica ideale del filtro avrebbero dovuto essere eliminate completamente. Questa banda di frequenza prende il nome di banda di transizione.
Per quantificare l'aumento di pendenza, si può osservare che passando ad un filtro del secondo ordine, la pendenza diventa di 40dB/decade, mentre in un filtro del quarto ordine è di 80dB/decade. Sostanzialmente si ha che i filtri di ordine alto sono ottenuti come la cascata di più filtri di ordine più basso.
La tipologia di filtro Butterworth passa basso è una fra le più utilizzate per effettuare uno smoothing. La sua risposta in frequenza è spesso nominata *maximally flat* (massimamente piatta), in quanto ha la proprietà di essere piatta in corrispondenza della componente continua (quella a frequenza zero) e in più, non ha oscillazioni fino alla frequenza di cut-off o frequenza di taglio. Il trade-off con questa proprietà è una banda di transizione piuttosto ampia.
Nel caso di questa tipologia di filtro conviene analizzare la sua risposta in frequenza, passando nel dominio della frequenza, tramite trasformata di Fourier. Questa è data dalla seguente equazione

$$H\left(j\omega\right) = \frac{1}{\sqrt{1+\varepsilon^2\left(\frac{\omega}{\omega_P}\right)^{2n}}}$$

Dove il parametro n è l'ordine del filtro, ε è il massimo guadagno della banda passante e ω_P è la frequenza di cut-off.

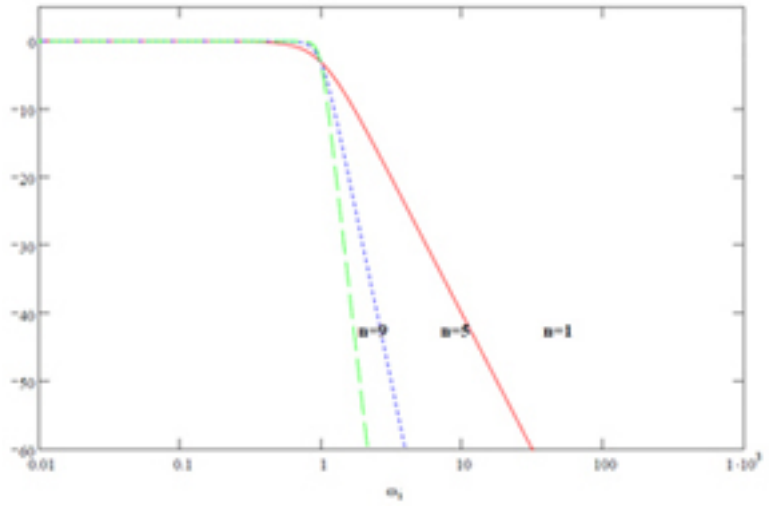
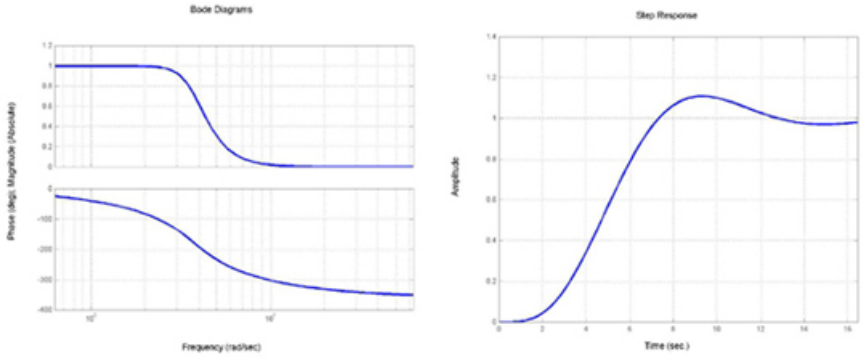


Figura 3.3: Risposta all'impulso di un filtro Butterworth al variare del grado del polinomio.



(a) (b)

Figura 3.4: Diagramma di Bode (figura (a)) e Risposta all'impulso (figura (b)) di un filtro Butterworth.

Il filtro Butterworth produce delle tabelle standardizzate di polinomi normalizzati, per filtri passa basso, come ad esempio quella mostrata di seguito.
I principali svantaggi di questa tipologia di filtro sono:

1. Dalla proprietà di massima piattezza, della risposta in frequenza, deriva un'esigenza di un ordine elevato, per ottenere una banda di transizione più stretta;

2. La sua risposta di fase non è lineare.

In particolare si ha che la massima piattezza corrisponde ad una risposta in frequenza, per cui tutte le derivate, fino a quella corrispettiva dell'ordine del

n	Normalised Denominator Polynomials in Factored Form
2	(1+1.414s+s2)
3	(1+s)(1+s+s2)
4	(1+0.765s+s2)(1+1.848s2)
5	(1+s)(1+0.618s+s2)(1+1.618s2)
6	(1+0.518s+s2)(1+1.414s+s2)(1+1.932s+s2)
7	(1+s)(1+0.445s+s2)(1+1.247s+s2)(1+1.802s+s2)
8	(1+0.390s+s2)(1+1.111s+s2)(1+1.663s+s2)(1+1.962s+s2)
9	(1+s)(1+0.347s+s2)(1+s+s2)(1+1.532s+s2)(1+1.879s+s2)
10	(1+0.313s+s2)(1+0.908s+s2)(1+1.414s+s2)(1+1.782s+s2)(1+1.975s+s2)

Tabella 3.1: Polinomi normalizzati per filtri Butterworth passa basso del secondo ordine.

filtro, risultano nulle in corrispondenza della frequenza continua, $\omega = 0$.

3.4 Filtro Chebyshev

Prima di passare a descrivere le caratteristiche di questo filtro è necessario introdurre il concetto di polinomio di Chebyshev. Questi hanno la seguente forma

$$C_n(\omega) = \cos(n \arccos(\omega)) \quad \omega \leq 1$$
$$C_n(\omega) = \cosh\left(n \cosh^{-1}(\omega)\right) \quad \omega > 1$$

Questo metodo è stato ideato e sviluppato dal matematico russo Pafnuti Chebyshev (1821-1894). Alla base di questo filtro c'è la strategia che prevede la presenza di oscillazioni nella banda passante per ottenere una banda di transizione molto ristretta. Se ad esempio, le oscillazioni sono annullate, il filtro assume la caratteristica di piattezza massimale, con effetti negativi sulla banda di transizione, come accade per il filtro Butterworth.
Nello specifico si parla di filtro di Chebyshev del primo tipo, nel caso in cui le oscillazioni siano ammesse solamente nella banda passante; si parla, invece, di filtro di Chebyshev del secondo tipo, quando le oscillazioni sono ammesse solo nella banda filtrata.
Per analizzare la funzione di trasferimento di questo filtro, è possibile prendere in esame il suo modulo quadro, preso nel dominio della frequenza, che avrà la seguente forma

$$|H(j\omega)|^2 = \frac{1}{1 + \varepsilon^2 C_n(\omega)^2}$$

in cui $C_n(\omega)$ è un polinomio di Chebyshev di grado n e ε imposta la dimensione delle oscillazioni.
Il filtro di Chebyshev rispetto a quello di Butterworth rilassa la richiesta in banda passante, per riuscire ad aumentare la pendenza nella banda di transizione. In questo modo si ha che, a parità di ordine del filtro e di parametri di tolleranza, si ottengono transizioni più ripide rispetto al filtro Butterworth.

3.5 Filtro Ellittico o Cauer

Un filtro ellittico, conosciuto anche come filtro Cauer, dal nome del matematico tedesco Wilhelm Cauer, è un filtro ad oscillazioni equalizzate (equiripple), presenti sia nella banda passante, sia nella banda bloccata. Il numero di oscillazioni in ognuna delle bande è regolabile in modo indipendente e nessun altro filtro dello stesso ordine ha un guadagno così alto, nella fase di transizione, che permette un'alta velocità di passaggio, da banda passante a banda filtrata, fissato il numero di oscillazioni. La possibilità di aggiustare le increspature può essere barattata, nella fase di design del filtro, con una intensità massimale alle variazioni.
Si ha che, più le oscillazioni nella banda filtrata si approssimano allo zero, più il filtro diventa simile a quello di Chebyshev del primo tipo. Se le oscillazioni si avvicinano allo zero nella banda passante, diventa un filtro di Chebyshev del secondo tipo. Se entrambe le oscillazioni si attenuano fino allo zero diviene un filtro Butterworth.
La funzione di trasferimento di un filtro passa basso ellittico ha la seguente descrizione

$$H\left(j\omega\right) = \frac{1}{\sqrt{1+\varepsilon^2 U_n^2(\xi\omega)}}$$

con $U_n(\omega)$ funzione ellittica Jacobiana, o funzione ellittica razionale, ε fattore d'oscillazione e ξ fattore di selettività. Nello specifico si ha che ε regola le oscillazioni nella banda passante ed un'opportuna combinazione di ε e di ξ regola le oscillazioni in banda filtrata.
Di seguito sono riportate alcune proprietà di questo filtro:

1. Nella banda passante la funzione ellittica razionale varia tra 0 e 1, mentre il guadagno in questa banda, varia tra 1 e $\frac{1}{\sqrt{1+\varepsilon^2}}$;
2. Nel caso in cui $\xi \to \infty$ la funzione ellittica razionale diventa simile ad un polinomio di Chebyshev e il filtro tende ad un filtro di Chebyshev del primo tipo;
3. I filtri ellittici hanno la risposta massimamente piatta per un dato ordine, ma risposta in fase estremamente non lineare.

3.6 Confronto dei Filtri

Per il test delle prestazioni dei filtri analizzati, sono stati utilizzati due dataset fortemente correlati tra loro. Questi sono stati costruiti sperimentalmente a partire dallo stesso modello: l'unica caratteristica che li differenzia è la presenza, in uno solo dei due, di una componente stocastica. La struttura di base di entrambi

i dataset è la seguente: 5 reti di 10 geni, campionati in 21 valori, per ognuna della quali sono presenti 4 set. Lo scopo del test è la capacità del filtro di ricostruire i dati originali, partendo dalla loro versione perturbata.
I filtri utilizzati per la comparazione sono i seguenti:

1. Filtro a media di 5 valori con finestra scorrevole;
2. Filtro con fitting lineare;
3. Filtro con fitting quadratico;
4. Filtro Savitzky-Golay;
5. Filtro Butterworth;
6. Filtro Chebyshev;
7. Filtro Ellittico.

Per ogni rete è calcolato l'errore medio relativo al singolo gene, al variare del set. I risultati ottenuti sono mostrati nella figura 3.5. Da questi si nota che la maggior parte dei filtri ha comportamento rassomigliante, in questo ambito d'analisi. In particolare, si ha che il filtro a finestra scorrevole è quello con le prestazioni peggiori.
Le prestazioni dei restanti filtri sono pressappoco dello stesso ordine. Quello che mediamente riesce a ricostruire con più precisione i dati di partenza è il filtro Savitzky-Golay. L'errore, nel caso dell'approssimazione di questo filtro, è di un ordine inferiore, rispetto a quello dei filtri rimanenti.

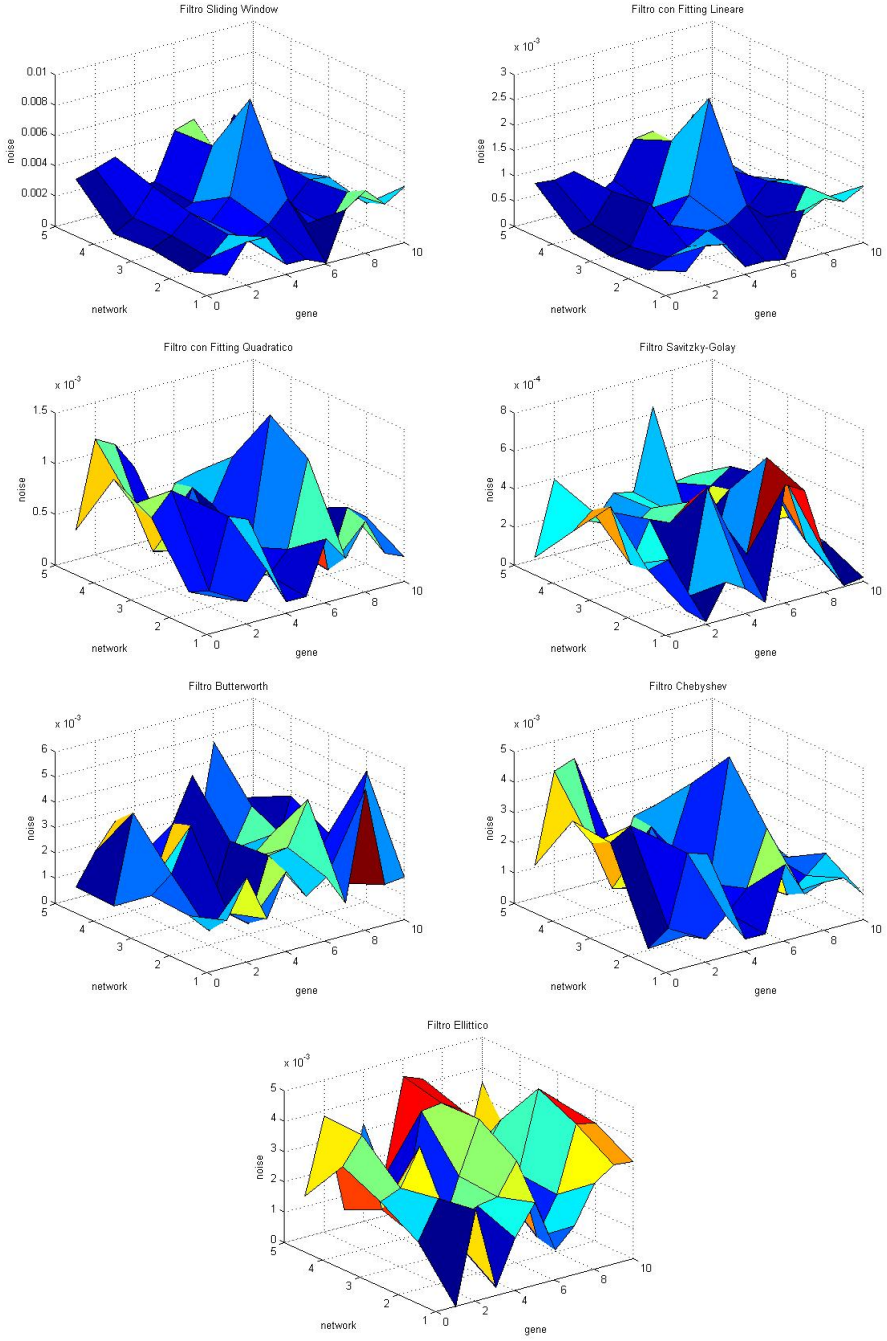


Figura 3.5: Grafici degli errori dei filtri nella ricostruzione dei dati originali a partire dalla loro versione perturbata.

Capitolo 4

Interpolazione

4.1 Introduzione

Il problema che si vuole affrontare in questa sezione è il seguente:
Data una funzione $F(x)$ continua, di cui si conoscono i valori $F(x_1), \ldots, F(x_n)$, corrispettivi ai punti $x_1, \ldots, x_n$, si vuole determinare i valori assunti dalla stessa, in corrispondenza dei punti $x_{i_1}, x_{i_2}, \ldots, x_{i_n}$, compresi nell'intervallo $[x_1\ x_n]$.

L'esigenza che porta alla formulazione di questo problema è il contesto in cui ci si trova a lavorare. I dati che si estrapolano dagli esperimenti, relativi ai profili genici, sono delle serie di valori reali. Nella cellula, si ha che questi andamenti variano come funzioni continue, per cui il processo di campionamento, introdotto con la rilevazione dei dati sperimentali, è un'ulteriore fonte di errori, nel processo globale di elaborazione.

L'obiettivo è quello di essere in grado di descrivere al meglio l'andamento della funzione sottostante ai dati discreti. Come sarà descritto con maggior dettaglio nel capitolo successivo, per il lavoro presentato in questa tesi, è importante identificare con precisione quei particolari punti, del profilo genico, che racchiudono in sé la maggior parte dell'informazione significativa. Un esempio immediato, di questa tipologia di punti significativi, sono i minimi e i massimi locali. La loro importanza sarà precisata meglio nel capitolo successivo, durante la descrizione delle rappresentazioni utilizzate per le espressioni geniche.

La condizione ideale di lavoro è quella di una descrizione continua della funzione, mentre la situazione reale di lavoro è un'insieme di serie campionate. A questo scopo l'interpolazione cerca di sopperire a questa mancanza, per poter estendere la numerosità dei campioni, che potranno essere utilizzati per l'analisi, in modo da ottenere un insieme sia più grande, sia più significativo.

Un esempio di procedura d'interpolazione è mostrata nell'immagine 4.1.

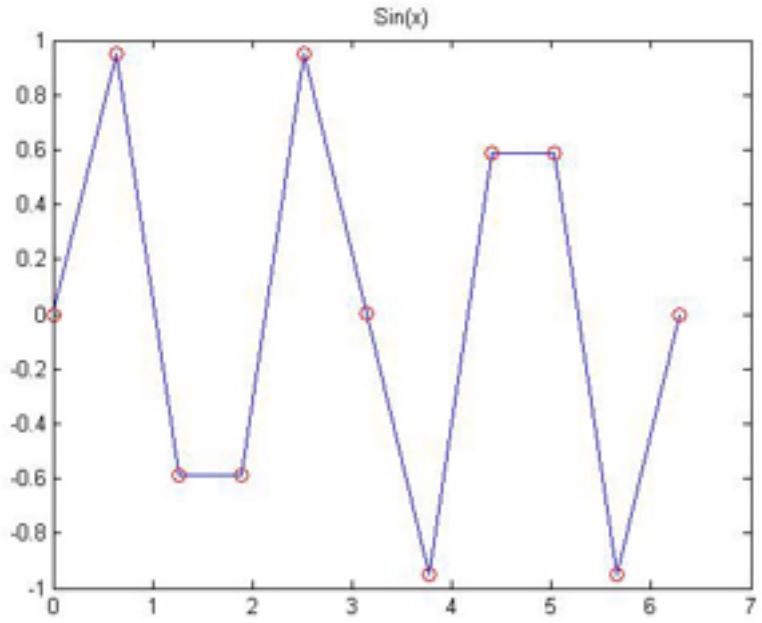


Figura 4.1: Interpolazione lineare di un segnale sinusoidale.

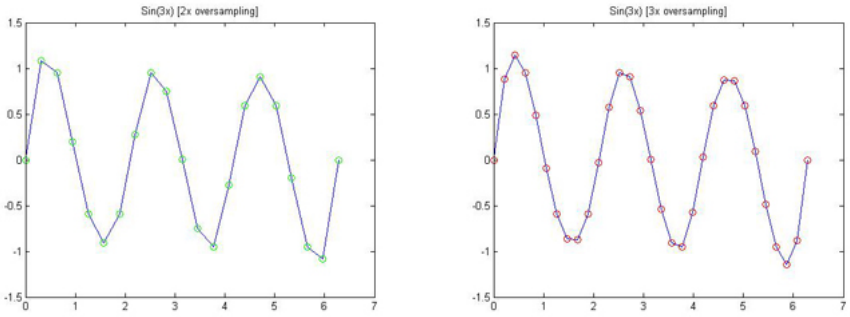
Il segnale preso in esame è $y(t) = \sin(3x)$ nel periodo $\begin{bmatrix} 0 & 2\pi \end{bmatrix}$. I cerchi sono i punti campionati, mentre la linea rappresenta il presunto andamento della funzione. Come si può notare, la distribuzione discreta degli istanti di campionamento ha introdotto un effetto indesiderato, vale a dire la perdita di alcune creste dell'onda sinusoidale.

Una possibile soluzione a questo problema è effettuare una sorta di oversampling, tramite interpolazione, ricavando un insieme più grande di valori da utilizzare nell'analisi. Il prezzo di questa procedura è l'aumento del costo computazionale del problema, in quanto si va ad aumentare la taglia dello stesso.

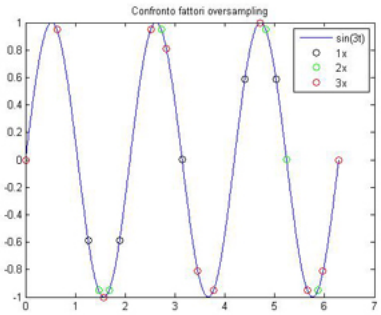
Utilizzare tutti questi nuovi dati, non è conveniente dal punto di vista computazionale, in quanto sarebbe un grosso fattore di rallentamento, ma dai dati interpolati è possibile ottenere un sottoinsieme di valori, avente taglia uguale, o comunque dello stesso ordine dei dati di partenza, ma che, in base ad un'opportuna scelta, riesce a descrivere maggiormente la funzione sottostante.

Come si nota dall'immagine 4.2, confrontando la serie originale con quelle ottenute tramite interpolazione si osserva che in entrambi i casi si ha una maggiore risoluzione delle creste delle sinusoidi.

L'interpolazione, essendo una tecnica di stima dell'andamento, ha un rendimento fortemente legato alla dinamica del segnale preso in esame. Nel caso precedentemente preso in esame, questo è una semplice sinusoide, non affetta da disturbo, per cui l'interpolazione fornisce una ricostruzione piuttosto fedele della



(a) (b)



(c)

Figura 4.2: Interpolazione polinomiale di un segnale sinusoidale, con un fattore di oversampling 3 (figura (a)) e 3 (figura (b)); confronto tra fattori di oversampling (figura (c)).

funzione. Nella realtà le distribuzioni, che devono essere analizzate, possono presentare anche andamenti molto meno regolari e quindi, le prestazioni dei metodi d'interpolazione possono peggiorare sensibilmente.
Dall'applicazione pratica delle tecniche d'interpolazione, su dati reali, si è ricavata una tendenza generale di questi metodi ad introdurre un maggiore scostamento, in corrispondenza delle due estremità del segnale, vale a dire nella porzione iniziale e in quella finale. Per risolvere questo problema è possibile introdurre un'opportuna finestra che assegna un determinato peso ai campioni, che risultano dall'interpolazione. Esistono diverse tipologie di finestre, le principali sono le seguenti:

1. Finestra trapezoidale, è contraddistinta da tre zone d'interesse, due laterali ed una centrale; la zona centrale definisce la porzione di segnale che non subirà nessuna distorsione; nelle zone laterali, la finestra introduce un'attenuazione lineare che compensa il generale scostamento dei dati interpolati nei margini del segnale;

2. Finestra basata su di una particolare distribuzione, a questa categoria appartengono le finestre sinusoidali e quelle basate sulla distribuzione normale; la scelta di una particolare distribuzione deve essere correlata con quello che è supposto essere l'effetto che si desidera introdurre sui dati da analizzare.

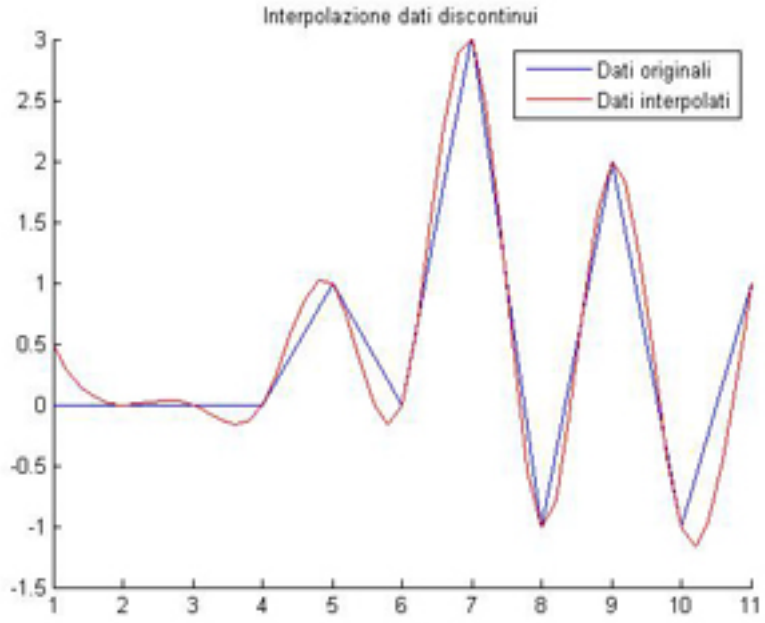


Figura 4.3: Confronto tra l'andamento reale di una funzione e quello ottenuto tramite interpolazione spline.

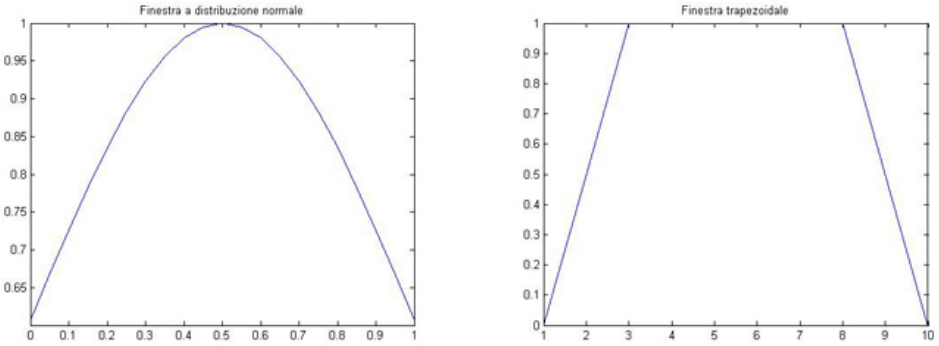
In generale si ha a che fare con una area centrale della distribuzione dei dati, su coi non è necessario effettuare alcuna operazione; per questa ragione, la finestra, in questa porzione, tenderà ad uno, seguendo la seguente funzione

$$F(x) = f(x) \cdot g(x)$$

in cui $f(x)$ è la funzione interpolata, $g(x)$ è la funzione della finestra ed infine $F(x)$ è la funzione risultante.
Si ha poi che più ci si allontana da questa zona centrale, più è necessario introdurre uno smorzamento, vale a dire che, verso i lati della finestra, la funzione $g(x)$ tenderà a decrescere. La scelta di una o di un'altra finestra determina principalmente le seguenti caratteristiche della finestra:

1. La larghezza della zona cosiddetta passante;

2. La rapidità con cui decade in corrispondenza delle zone laterali.



(a) (b)

Figura 4.4: Tipologie di finestre. trapezoidale (figura (a)) e normale (figura (b)).

4.2 Interpolazione di Newton

Il problema dell'interpolazione può essere sintetizzato con il seguente inciso:
Di una funzione incognita $F(x)$ si conoscono i valori $y_0, y_1, \ldots, y_k$, che essa assume in $k+1$ punti distinti, $x_0, x_1, \ldots, x_k$ e si vogliono i valori di $F(x)$ in punti intermedi tra quelli in cui essa è nota.
Tale problema risulta indeterminato, in quanto è possibile fissare un numero infinito di funzioni che passano per i $k+1$ punti, assunti dalla funzione $F(x)$, ma che hanno punti intermedi differenti. Il problema diventa determinato quando si sostituisce ad $F(x)$ una funzione appartenente ad una classe di funzioni, dipendenti da un certo numero di parametri, che sono unici, una volta fissate le condizioni

$$f(x_i) = y_i \quad i = 0, 1, \ldots, k$$

La funzione $f(x)$ è detta interpolante e ricavarla significa risolvere il problema dell'interpolazione, nella classe delle funzioni considerate. Supponiamo che questa classe sia costituita, ad esempio, dalla classe dei polinomi di grado non maggiore a k. Si parla in tal caso d'interpolazione parabolica.
Una formula che risolve il problema dell'interpolazione parabolica è

$$f(x_0) = F(x_0) + (x - x_0) F(x_0, x_1) + (x - x_0)(x - x_1) F(x_0, x_1, x_2) + \ldots$$

$$\cdots + (x - x_0)(x - x_1) \ldots (x - x_{k-1}) F(x_0, x_1, \ldots x_k)$$

Detta anche formula d'interpolazione di Newton.
A questo punto è necessario introdurre il concetto di differenza divisa, in quanto i coefficienti del polinomio al secondo membro si ottengono tramite questa tecnica.
Data una funzione $f(x)$, definita nell'intervallo $[a, b]$ ed una successione di punti $x_0, x_1, \ldots, x_n, \ldots$, distinti e appartenenti all'intervallo $[a, b]$, si definisce differenza divisa prima (o del primo ordine) la funzione

$$f(x_{r_0}, x) = \frac{f(x) - f(x_{r_0})}{x - x_{r_0}}$$

Definita per $x \neq x_{r_0}$ in $[a, b]$.
Si definisce differenza divisa seconda (o del secondo ordine) la funzione

$$f(x_{r_0}, x_{r_1}, x) = \frac{f(x_{r_0}, x) - f(x_{r_0}, x_{r_1})}{x - x_{r_1}}$$

Definita per $x \neq x_{r_0}, x_{r_1}$ in $[a, b]$.
In generale si chiama differenza divisa $(k+1) - esima$ (o di ordine $k+1$) la funzione

$$f(x_{r_0}, x_{r_1}, \ldots, x_{r_k}, x) = \frac{f(x_{r_0}, \ldots, x_{r_{k-1}}, x) - f(x_{r_0}, , \ldots, x_{r_k})}{x - x_{r_k}}$$

Definita per $x \neq x_{r_0}, x_{r_1}, \ldots, x_{r_k}$ in $[a, b]$.
Partendo da queste definizioni è possibile dimostrare che la funzione così ottenuta passa per tutti i $k+1$ punti e che questa funzione è unica.
Ritornando alla tavola delle differenze divise si possono utilizzare differenti notazioni; di seguito ne è riportata una delle possibili.

x	**F(x)**	**d.d. I° ordine**	**d.d. II° ordine**	**d.d. III° ordine**
x0	F(x0)			
x1	F(x1)	F(x0,x1)		
x2	F(x2)	F(x0,x2)	F(x0,x1,x2)	
x3	F(x3)	F(x0,x3)	F(x1,x2,x3)	F(x0,x1,x2,x3)
...	...	...	...	...

Tabella 4.1: Tabella delle differenze divise, per il calcolo dei coefficienti del polinomio d'interpolazione di Newton.

4.3 Spline

Dallo studio dell'interpolazione, tramite la classe dei polinomi, si è osservato che la funzione è approssimata bene nella parte centrale dell'intervallo d'interpolazione, mentre agli estremi, il risultato è sensibilmente peggiore. Questo comportamento sembra accentuarsi anche con l'aumento dei punti d'interpolazione.
Si è indotti quindi a pensare che il principale inconveniente della classe dei polinomi, sia quello di essere poco flessibile. I polinomi sembrano andare bene su intervalli sufficientemente piccoli, ma quando si passa ad intervalli più larghi, sono introdotte spesso delle oscillazioni eccessive. Questo suggerisce che, allo scopo di ottenere una classe di funzioni di approssimazione con maggiore flessibilità, si deve lavorare con polinomi di grado relativamente basso e dividere l'intervallo interessato in più sottointervalli.

Con questo scopo sono state introdotte le spline. Sostanzialmente si tratta di curve che uniscono, nel modo più regolare possibile, punti di coordinate (x_i, y_i) assegnate. Queste sono particolarmente adatte per l'approssimazione numerica di equazioni differenziali e di equazioni integrali.
Prima di iniziare a descrivere le caratteristiche delle spline è necessario introdurre il concetto di polinomio a tratti, che sopperisce ai limiti imposti dai polinomi regolari.
Dato un intervallo $[a, b]$ e fissata una partizione di $k+1$ punti, detti nodi, che lo suddivide in k sottointervalli I, del tipo

$$a = x_0 < x_1 < \cdots < x_k = b, I_i = (x_i, x_{i+1})\, i = 0, 1, \ldots, k-1$$

Fissate le funzioni polinomiali $P_0, P_1, \ldots, P_k$, con $k \leq m$, si definisce

$$P_m(\triangle) = \{f(x) \mid f(x) = P_j \; con \; x \in I_i \;, \quad i = 0, 1, \ldots, k-1; 0 \leq j \leq k; k \leq m\}$$

Mentre è chiaro che si è ottenuta maggiore flessibilità, passando dai polinomi regolari ai polinomi a tratti, è venuta a mancare un'altra importante proprietà: le funzioni polinomiali a tratti, in generale, non sono regolari.

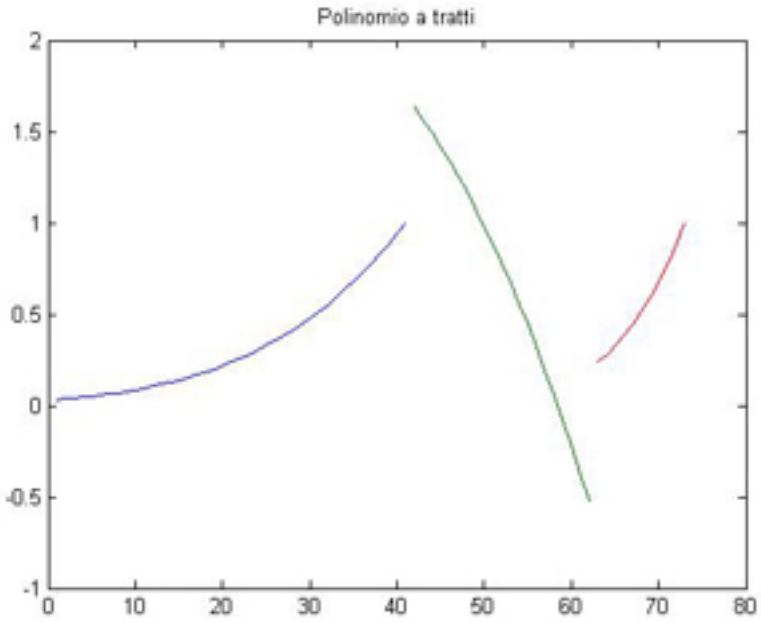


Figura 4.5: Esempio di polinomi a tratti.

A questo scopo è introdotta la seguente classe di funzioni:
Fissato $\triangle$, come nella definizione precedente, sia $m \epsilon N$; se si indica con $C^{m-1}[a, b]$ la classe delle funzioni continue, con derivata fino all'ordine $(m-1) - esimo$

continua nell'intervallo $[a, b]$ allora

$$S_m(\triangle) = P_m(\triangle) \cap C^{m-1}[a, b]$$

indica la classe delle spline polinomiali di grado m sulla partizione $\triangle$.
$S_m(\triangle)$ restringe così la classe dei polinomi a tratti, imponendo che le funzioni appartenenti ad essa siano dei polinomi a tratti continui e con derivata fino all'ordine $(m-1) - esimo$ continua. Una determinata spline è interpolante di una funzione incognita, se risultano verificate contemporaneamente tutte le condizioni iniziali,

$$S_\triangle(x_i) = f(x_i) \qquad\qquad i = 0, 1, \ldots, k$$

Nella pratica le spline risultano più efficienti dell'interpolazione, basata sulla classe dei polinomi, per cui sono usate maggiormente come metodo d'interpolazione.

Capitolo 5

Tecniche per l'Analisi dei Dati di Genomica

5.1 Introduzione

Scopo di questo capitolo è descrivere quali siano stati gli obiettivi che si sono prefissati, nell'ambito dell'analisi dei dati di genomica e come si sia cercato di risolverli. A questo punto, sono stati già introdotti i concetti riguardanti una prima parziale elaborazione di questi dati, in modo da renderli più efficienti, nei successivi passi di analisi. Quello con cui si ha a che fare sono delle serie temporali, corrispondenti a degli andamenti delle concentrazioni dei geni. Essendo la quantità di dati, ottenuta tramite DNA-microarray, notevole, risulta necessario un procedimento per l'estrapolazione dell'"*informazione significativa*".
L'ambiente in cui ci si trova a lavorare, suggerisce la possibilità di fare ricorso a tecniche legate ai processi di apprendimento automatico e di classificazione. Per esempio, in ambito medico e biologico, possono essere utilizzate tecniche di data mining e di Machine Learning per la raccolta, la classificazione e la successiva elaborazione di profili di pazienti, che presentano o meno una determinata malattia, in modo da ottenere un criterio per il riconoscimento della stessa patologia, su nuovi pazienti. Esempi di articoli in quest'ambito sono [LIKG93, Fuzzy pattern recognition, 1993], tecniche di Machine Learning in ambito diagnostico, [BDC05, Microarray time series, 2005], per l'analisi dei microarray. Sono presenti anche molte tesi, incentrate su questo settore, come ad esempio [Gre09, Geni differenzialmente espressi, 2009], tecniche per la classificazione di geni differentemente espressi, [Mar08, Modelli Grafici per l'Analisi di Network Biologici, 2008], combinazione di modelli grafici, [Ris08, Analisi di dati di espressione genica, 2008], effetto della normalizzazione sull'analisi statistica dei dati genici, [Gar03, Studio delle Leucemie, 2007], studio delle leucemie.

Il lavoro di Sacchi e al. [LS05] si inoltra più nello specifico nel campo da noi preso in esame e rappresenta una sorta di base di partenza per il lavoro descritto in questa tesi. L'obiettivo della ricerca di Sacchi è quello di ricostruire le reti di regolazione genica, ovvero determinare le relazioni d'interazioni tra geni. Nello specifico, il problema affrontato, in questo caso, è quello della determinazione del rapporto di causalità, in una coppia di geni, di cui si conosce a priori l'esistenza di un rapporto di dipendenza; se si guarda questo problema dal punto di vista della rete di regolazione, esso può essere espresso altrimenti come la ricerca del verso dell'arco che collega due geni, di cui si conosce a priori il rapporto d'interdipendenza.
Le principali tecniche utilizzate appartengono sostanzialmente alla categoria degli algoritmi per il Machine Learning. Per una panoramica di queste tecniche e per la descrizioni di alcune di esse, in particolare di quelle a cui si è fatto ricorso durante l'intero lavoro, si rimanda all'appendice A.
Come prefissato all'inizio della tesi, gli obiettivi che sono stati posti sono due e sono i seguenti:

1. Ricostruzione dell'intera rete di regolazione genica;
2. Ricerca di sottostrutture significative (*FFL*) presenti nella rete di regolazione genica.

Il filo conduttore, di entrambi questi scopi, è la comparazione di due rappresentazioni, utilizzate come base per la descrizione degli andamenti dei geni, vale a dire il confronto tra una rappresentazione numerica ed una rappresentazione simbolica.
La prima è la descrizione immediata che si ha dei dati, in quanto si fa ricorso semplicemente alle effettive serie temporali, ottenute dagli esperimenti. La seconda rappresentazione è basata sull'idea formulata da Falda nel suo lavoro [Fal09]. Quest'articolo, insieme a quello di Sacchi et al. costituisce l'intero punto di partenza di questa tesi. L'articolo di Falda, definisce una rappresentazione, denominata simbolica, dal fatto che si è ricorso ad un opportuno alfabeto di simboli, utilizzati per descrivere qualitativamente i profili genici. Gli elementi dell'alfabeto corrispondono alle configurazioni significative ricercate dei segnali; queste, che prendono il nome di astrazioni temporali, possono essere semplici, quando corrispondono ad un singolo punto d'interesse del profilo, oppure complesse, se corrispondono invece a degli andamenti, vale a dire all'unione di due o più astrazioni temporali semplici. A questo punto il confronto tra i geni si effettua tramite opportune metriche, basate sui simboli dell'alfabeto scelto, invece che sui profili numerici. Nel capitolo seguente sarà spiegata con maggiore dettaglio proprio questa rappresentazione.
La caratteristica principale delle serie, relative agli andamenti dei geni, è il fattore tempo. Si tratta, infatti, di serie temporali, vale a dire serie costituite da

coppie di dati $\left\langle \begin{array}{cc} t_i & v_i \end{array} \right\rangle$ in cui il parametro t_i specifica un preciso istante temporale, mentre v_i è il valore di concentrazione del gene in esame nell'istante t_i. Osservando quest'evoluzione temporale, ci si accorge sperimentalmente che è solo una parte del segnale complessivo che costituisce la sua componente significativa. Un'altra nota, che nasce dall'analisi sperimentale, è che, avendo a che fare con serie temporali, l'utilizzo di metriche classiche di rassomiglianza tra funzioni, come ad esempio la distanza euclidea, porta generalmente a risultati scadenti.
Da queste osservazioni è nata l'idea di ricorrere ad una rappresentazione differente, tramite opportuni simboli, per utilizzare nell'analisi di correlazione, solo le porzioni significative dei segnali.
Per quanto riguarda, invece, la rilevazione delle sottostrutture FFL, presenti nelle reti di regolazione, si è utilizzata un'altra famiglia di algoritmi: le SVM.
Al termine di questo capitolo saranno definiti i concetti utilizzati per la valutazione dei processi di reperimento dell'informazione, nei casi presi in esame, per poter confrontare tra loro i modelli prodotti.

5.2 Rappresentazione Simbolica

La base di partenza da cui partire sono le serie numeriche. Ciascun profilo del singolo gene avrà la forma $P_i = \left\langle \begin{array}{cc} t_i & v_i \end{array} \right\rangle$, in cui il valore assunto dalla concentrazione del gene (parametro v_i) è individuato in un determinato istante temporale (parametro t_i).
A questo punto. a partire dalla serie di campioni temporali $P_1, \ldots, P_n$, ci si prefigge l'obiettivo di ottenere un'astrazione temporale, che costituisce una descrizione qualitativa del profilo del gene. Procedendo in questo modo si ha una effettiva perdita d'informazione, ma quello che ci si prefigge è che sia comunque mantenuta la porzione significativa del segnale.
L'astrazione temporale da ottenere sarà costituita da una sequenza di elementi (simboli) aventi la seguente forma

$$\langle s_i,\ t_{i_{START}},\ t_{i_{STOP}}, x_i \rangle$$

in cui i parametri hanno il seguente significato:

1. s_i è un simbolo che identifica univocamente un particolare pattern rilevante, in pratica sono i simboli elementari dell'alfabeto;
2. $t_{i_{START}}$ è l'istante iniziale del pattern d'interesse;
3. $t_{i_{STOP}}$ è l'istante finale del pattern d'interesse;
4. x_i è un parametro aggiuntivo che quantifica l'entità del pattern d'interesse.

Una volta fissato l'alfabeto della rappresentazione simbolica, in modo da definire quali sono gli andamenti d'interesse che devono essere riconosciuti, e fissata la loro codifica, è necessario effettuare un passo di etichettatura, vale a dire un'assegnazione di questi simboli ai profili dei geni presi in esame. Questa procedura si compone di due passi:

1. Determinazione dei punti dominanti tramite algoritmo CAL;
2. Determinazione delle astrazioni temporali.

Un metodo per ricavare i punti dominanti è utilizzare un'approssimazione lineare, per mezzo di spezzate. Gli algoritmi che svolgono questo scopo possono essere raggruppati in tre tipologie:

1. Algoritmi *sliding window*, che estendono ricorsivamente dei segmenti di segnale, continuando ad aggiungere punti fintantoché non è superata una soglia d'errore prefissata;
2. Algoritmi *top-down*, che partizionano ricorsivamente le serie temporali, finché non si verifica una condizione d'arresto;
3. Algoritmi *bottom-up*, che partendo dall'approssimazione migliore, fondono iterativamente segmenti contigui, finché non si verifica una condizione di arresto.

L'algoritmo CAL (Chord and Length), ideato per applicazioni nell'ambito del filtraggio d'immagini, è di tipo sliding window. Per comprendere meglio il suo funzionamento è necessario introdurre alcune definizioni.

Definizione 5.1 (corda)**.** *Data una coppia di punti temporali* $P_i = \left\langle \begin{array}{cc} t_i & v_i \end{array} \right\rangle$ *e* $P_j = \left\langle \begin{array}{cc} t_j & v_j \end{array} \right\rangle$ *si definisce corda tra i punti* P_i *e* P_j *la lunghezza del segmento* $C_{i,j}$ *che li congiunge e si ottiene come la distanza euclidea tra i due punti*

$$C_{i,j} = \sqrt{(t_j - t_i)^2 + (v_j - v_i)^2}$$

Definizione 5.2 (arco)**.** *Data una coppia ordinata di punti temporali* $P_i = \left\langle \begin{array}{cc} t_i & v_i \end{array} \right\rangle$ *e* $P_j = \left\langle \begin{array}{cc} t_j & v_j \end{array} \right\rangle$*, sia* $P_{i+1}, P_{i+2}, \ldots, P_{j-1}$ *la sequenza ordinata dei punti compresi tra* P_i *e* P_j*; si definisce arco tra i punti* P_i *e* P_j *la somma delle lunghezze delle corde tra tutte le coppie di punti temporali consecutive comprese tra loro, e si calcola come*

$$A_{i,j} = \sum_{k=i}^{j-1} C_{k,k+1}$$

Definizione 5.3 (pendenza)**.** *Dati due punti dominanti consecutivi* $P_i = \left\langle \begin{array}{cc} t_i & v_i \end{array} \right\rangle$ *e* $P_j = \left\langle \begin{array}{cc} t_j & v_j \end{array} \right\rangle$*, sia* $\triangle_{v_{i,j}} = v_j - v_i$ *e* $\triangle_{t_{i,j}} = t_j - t_i$ *e sia* $C_{i,j}$ *la lunghezza della corda che li congiunge; la pendenza fra questi due punti è data dall'espressione*

$$p_{i,j} = \frac{\triangle_{v_{i,j}}}{\triangle_{t_{i,j}}}$$

Nel caso dell'algoritmo in esame si è fatto ricorso ad una versione leggermente differente della definizione di pendenza, vale a dire ad una pendenza approssimata, calcolata come

$$\sin(\alpha) = \frac{\triangle_{v_{i,j}}}{C_{i,j}}$$

In questo modo si ottiene una mappatura, di tutte le possibili pendenze, nell'intervallo $\left[\begin{array}{cc} -1 & 1 \end{array} \right]$.
Ritornando al procedimento generale, per ottenere la rappresentazione simbolica, si inizia utilizzando una funzione monotona crescente, per individuare i punti temporali nei quali si ha un cambiamento significativo della pendenza. Dopo di ciò, bisogna determinare un'approssimazione della serie temporale, mediante spezzate, considerando solo i punti in cui si verifica un cambiamento significativo della pendenza. A questo scopo sono calcolati i punti dominanti, ovvero un sottoinsieme dei punti iniziali, in cui la pendenza subisce una variazione significativa, ovvero maggiore di una soglia prefissata.
Dal punto di vista computazionale, il primo e l'ultimo punto sono considerati sempre come dei punti dominanti. Partendo dal primo, poi si calcola il valore della corda tra la coppia corrente di punti(quello attuale e l'ultimo visionato) e l'arco tra il punto corrente e l'ultimo punto dominante trovato. In aggiunta si ricava un coefficiente denominato T, definito come

$$T = \frac{\sqrt{A^2 + C^2}}{2}$$

Se questo risulta maggiore di una soglia prefissata il punto è definito come dominante. Nella pratica più questa soglia è fissata vicino allo zero, maggiore sarà il numero di punti riconosciuti come dominanti, mentre, aumentandola si ottiene una maggiore selettività. A questo punto è possibile ottenere le astrazioni temporali, nella forma $\langle s_i,\ t_{i_{START}},\ t_{i_{STOP}}, x_i \rangle$. Nello specifico, il parametro s_i è un carattere e l'unione di tutti questi simboli forma l'alfabeto dalla rappresentazione. Nel nostro caso in esame, l'alfabeto utilizzato è quello definito da Falda, nel suo articolo, e comprende i seguenti simboli:

1. Crescente, utilizzato principalmente per i punti di confine della serie ed indica una pendenza positiva del segnale;

2. Decrescente, come per il precedente è utilizzato dai punti di confine ed indica una pendenza negativa;

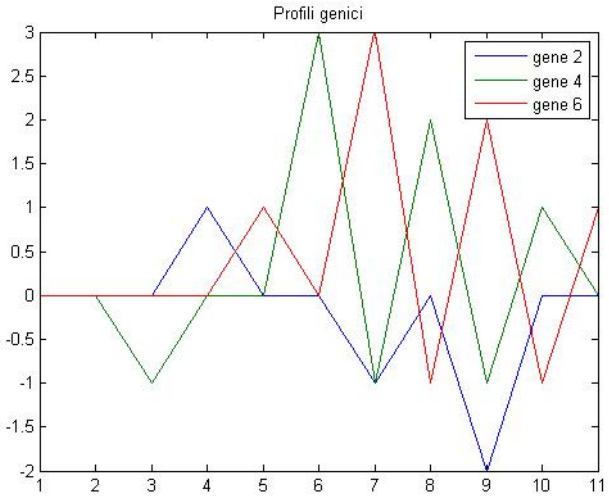


Figura 5.1: Profili di tre geni generati artificialmente.

3. Massimo, punto di passaggio da una pendenza positiva ad una negativa;
4. Minimo, punto di passaggio da una pendenza negativa ad una positiva;
5. Flesso, punto di cambiamento di segno della derivata seconda del segnale;
6. Stazionario, porzione del segnale, sufficientemente lunga, in cui esso non subisce sostanziali variazioni;
7. Zero, punto stazionario in cui il valore assoluto, assunto dal segnale è minore di una soglia, prossima allo zero;
8. Saturazione, punto stazionario in cui il valore assoluto assunto dal segnale è distante meno di un ϵ, prossimo allo zero, dal valore massimo assunto dal segnale.

Nella figura 5.1 sono riportati alcuni andamenti, utilizzati per la fase di test degli algoritmi implementati. I geni sono descritti da serie temporali, costituite da 11 campioni. Le rappresentazioni simboliche ottenute dall'algoritmo sono le seguenti:

1. Gene 2: zero, stazionario, massimo, flesso, stazionario, minimo, massimo, minimo, flesso, stazionario;
2. Gene 4: minimo, stazionario, minimo, flesso, stazionario, massimo, minimo, massimo, minimo, massimo, decrescente;
3. Gene 6: zero, stazionario, massimo, minimo, massimo, minimo, massimo, minimo, crescente.

A questo punto si ha a disposizione delle stringhe, costituite da simboli, appartenenti all'alfabeto e fortemente correlate agli andamenti dei geni analizzati. Per la misura effettiva d'interdipendenza tra queste stringhe sono stati utilizzati gli algoritmi LCS-CAL (Longest Common Subsequence - Chord and Length) e LCString-CAL (Longest Common String - Chord and Length).

Si ricorda che per sottosequenza di una stringa $x = x_1, \ldots, x_n$ si definisce una sequenza $z = z_1, \ldots, z_k$, con $k < n$, se esiste un insieme ordinato d'indici strettamente crescente, $i_1, \ldots, i_k$, tale che $\forall j = 1, \ldots, k$ si ha che $x_{i_j} = z_j$.

Il problema della determinazione della sottosequenza comune di lunghezza maggiore si risolve tramite programmazione dinamica. L'approccio basato sul brute-force, vale a dire provando tutte le possibili sottosequenze delle stringhe, non è percorribile, perché la sua complessità computazionale è esponenziale e quindi non praticabile anche per brevi sequenze di simboli. Il problema della sottosequenza comune di lunghezza comune gode della proprietà di sottostruttura ottima, vale a dire che una LCS di due sequenze contiene al suo interno una LCS dei prefissi delle due sequenze. Grazie a questa proprietà è possibile risolvere il problema ricorsivamente, con un algoritmo basato sulla programmazione dinamica.

Il problema si pone nei seguenti termini: date due sequenze, $x = x_1, \ldots, x_n$ e $z = z_1, \ldots, z_m$, si crea una matrice C, in cui il valore $C[i, j]$ è la lunghezza della LCS tra X_i e Z_j, dove X_i e Z_j sono i prefissi delle due rispettive sequenze. Si inizia il calcolo partendo da $C[i,\ 0] = 0\ \forall i = 1, \ldots, n$ e $C[0, j] = 0\ \forall j = 1, \ldots, m$ che costituiscono i casi base. Grazie alla proprietà di sottostruttura ottima si definiscono i restanti valori della matrice secondo la seguente formula

$$C[i, j] = \begin{cases} C[i-1, j-1] + 1 & se\ x_i = z_j \\ \max(C[i-1, j], C[i, j-1]) & sex_i \neq z_j \end{cases}$$

Dal momento che ci sono solo $\theta(nm)$ possibili sottoproblemi distinti, è possibile utilizzare la programmazione dinamica per il calcolo di tipo bottom-up. La matrice è ricavata in ordine raw major, dopo essere stata inizializzata con i casi base. Ad ogni passo è memorizzato anche quale dei sottoproblemi è risultato ottimo, in modo da ridurre successivamente i calcoli per la ricostruzione della soluzione ottima. In questo modo si ha che il tempo complessivo di esecuzione dell'algoritmo è $O(nm)$.

Fino a questo punto si è considerata solo la relazione di uguaglianza tra simboli che, riferito alla rete di regolazione genica, indica una relazione positiva tra due geni. Con lo stesso procedimento è possibile ricostruire anche le regolazioni negative. Mentre nel caso di regolazione positiva si parla di LCS diretta, nel caso contrario si parla di LCS inversa. La funzione di uguaglianza utilizzata in quest'ultimo caso non è una semplice negazione della versione precedente: due simboli sono considerati relazionati in maniera negativa se corrispondono ad

una delle seguenti coppie, considerate senza l'ordinamento, (massimo, minimo), (crescente, decrescente), (zero, saturazione), (flesso, flesso).

A questo punto per decidere se due geni sono tra di loro correlati si confronta la lunghezza della LCS reciproca, con un valore di soglia.

Per quanto riguarda l'algoritmo LCString, tutto il procedimento fin qui descritto, corrisponde, fatta eccezione l'uso delle sottosequenze. Questo algoritmo, invece, calcola la sottostringa comune, per cui le corrispondenti sottosequenze calcolate devono essere costituite da elementi consecutivi della stringa di partenza. Formalmente si ha, quindi, che la differenza consiste nell'imposizione che l'insieme d'indici, oltre ad essere strettamente crescente, deve anche essere costituito da elementi consecutivi. Per quanto riguarda gli altri aspetti, fin qui mostrati, con questo approccio si procede esattamente come accadeva nel caso precedente, calcolando la matrice tramite programmazione dinamica.

Questi metodi permettono di determinare, oltre alla presenza di correlazione tra due geni, anche il rapporto di causalità, vale a dire quale gene regola l'altro, oltre anche al tipo di regolazione, positiva o negativa, grazie all'uso della comparazione diretta o inversa.

5.3 Modifica Fuzzy della Rappresentazione Simbolica

Il problema che emerge, utilizzando direttamente i simboli nella misura di correlazione tra i profili di due geni, è che questa rappresentazione non contiene sufficiente informazione, per ricostruire al meglio la rete di regolazione. Sperimentalmente, infatti, questa rappresentazione simbolica, utilizzata su dataset reali, fornisce buoni risultati per determinare il verso della regolazione tra due geni, di cui si conosce a priori il rapporto di correlazione. Per quanto riguarda il problema generale e più complesso, di cercare di ricostruire l'intera rete, le prestazioni ottenute sono inferiori di quelle ricavate con l'uso degli algoritmi ARACNE e Reveal, che rappresentano l'effettivo stato dell'arte in questo campo.

Per questa ragione si è deciso di introdurre un uso più approfondito delle componenti delle astrazioni temporali. Ricordando che esse si presentano nella forma $\langle s_i,\ t_{i_{START}},\ t_{i_{STOP}}, x_i\rangle$, come precedentemente descritto, si è deciso di introdurre tutti i parametri all'interno della procedura di estrazione della sottosequenza o sottostringa comune. In precedenza, si aveva, infatti, che i dati descrittivi dell'astrazione temporale venivano utilizzati solo nella fase di etichettatura, vale a dire come discriminante per l'elemento dell'alfabeto da utilizzare.

La possibilità di utilizzare tutte questi dati aggiuntivi, anche nella fase successiva all'etichettatura, si sfrutta introducendo alcuni concetti appartenenti alla logica fuzzy, all'interno della procedura del calcolo della misura di correlazione fra i profili genetici.

L'idea di base è quella di aggiungere, ai simboli dell'alfabeto, dell'informazione di natura fuzzy: il concetto espresso dal simbolo massimo, sarà ad esempio, affiancato da un valore continuo (compreso tra 0 e 1), tarato sull'effettiva altezza relativa del picco del segnale e che costituirà una descrizione aggiuntiva, da utilizzare nella fase di analisi. Si va così a costituire un insieme fuzzy, che arricchirà la quantità complessiva d'informazione, che sarà possibile utilizzare. Per comprendere al meglio il procedimento adottato è necessario introdurre alcune definizioni appartenenti alla logica fuzzy.

Definizione 5.4 (funzione d'appartenenza)**.** *Un insieme fuzzy A è definito da una funzione di appartenenza (membership function)*
$\mu_A : X \in [0,1]$ *in cui X è l'universo di definizione, vale a dire un opportuno insieme.*

Dalla logica fuzzy derivano le seguenti misure, riferite ad specifica funzione di appartenenza:

1. $supporto = \{x \in X | \mu_A(x) > 0\}$;
2. $core = \{x \in X | \mu_A(x) = 1\}$;
3. $altezza = \sup_{x \in X} \mu_A(x)$.

In base alle caratteristiche di questi parametri è possibile definire diverse categorie di funzioni d'appartenenza. Il passaggio alla logica fuzzy impone la definizione di nuovi operatori, che sostituiranno quelli dell'algebra, comunemente usati. L'unione di due insiemi fuzzy è definita dalla seguente equazione

$$\mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x))$$

L'intersezione sarà definita come

$$\mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x))$$

Infine, il complemento di un insieme fuzzy è definito come

$$\mu_{\overline{A}} = 1 - \mu_A$$

Utilizzare la logica fuzzy in questo contesto rende possibile una comparazione tra andamenti più dettagliata. Nell'estrazione delle sottostringhe o sottosequenze comuni è necessario quantificare una correlazione tra simboli. Prendiamo ad esempio la seguente figura.
Il segnale che rappresenta il profilo del gene Y è una replica traslata nel tempo ed attenuata del profilo del gene X. Da questo ne deriva che il gene X regola positivamente il gene Y. Se però prendiamo ad esempio il profilo del gene Z, si ha

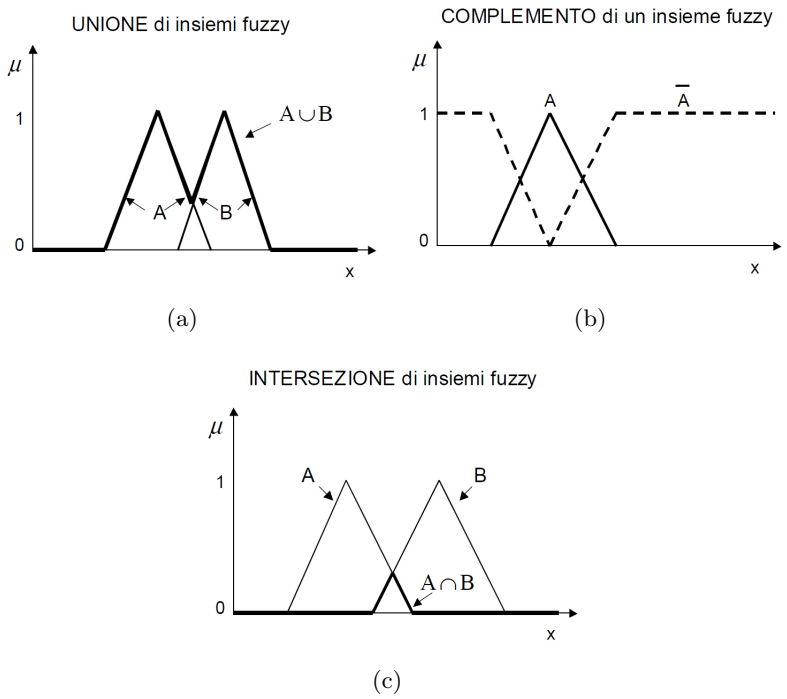


Figura 5.2: Operazioni tra insiemi fuzzy.

una situazione di indecisione. Dal punto di vista della rappresentazione simbolica, si ha una successione di simboli equivalenti per tutti e tre i profili (minimo, flesso, massimo, flesso, minimo, flesso, massimo). Quello che dovrebbe far decidere sulla presunta dipendenza tra il gene X e il gene Z è l'entità di questi simboli, o meglio la relazione tra le entità dei simboli di un profilo. Dal punto di vista linguistico, un agente umano osserverebbe che il gene X ha una coppia di punti dominanti di altezza relativa piccola, seguiti da una coppia avente altezza relativa grande. Questa caratteristica corrisponde nel profilo del gene Y, mentre non è presente nel gene Z, in cui si ha una sequenza di punti dominanti in cui l'altezza relativa rimane costante.

La logica fuzzy permette, tramite un insieme di funzioni di appartenenza, ognuna delle quali è riferita ad uno specifico elemento dell'alfabeto, di quantizzare questa appartenenza, ottenendo una descrizione aggiuntiva che, nel campo della dialettica, si avrebbe aggiungendo degli aggettivi (un minimo basso, un massimo grande).

L'intero processo per l'ottenimento di questa rappresentazione, denominata fuzzy, è costituito dai seguenti passi:

1. Definizione dell'insieme delle funzioni d'appartenenza ai simboli dell'alfabeto utilizzati;

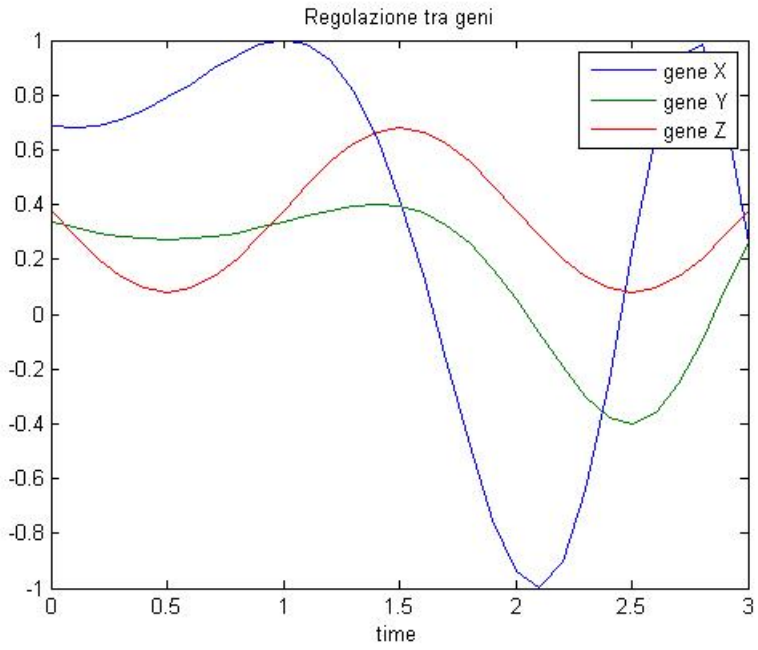


Figura 5.3: Esempio di una falsa correlazione tra tre geni.

2. Determinazione delle astrazioni temporali, come il sottoinsieme maggiormente significativo dei punti dominanti;

3. Assegnazione, ad ognuna delle astrazioni temporali, del valore fuzzy di appartenenza del punto al simbolo relativo;

4. Estrazione della sottostringa o sottosequenza comune, che presenta il rapporto di appartenenza ai simboli migliore.

Passiamo ora ad analizzare in maggiore dettaglio ogni singolo passo del procedimento. Nella fase preliminare si procede con la definizione dell'insieme dei simboli, utilizzati come alfabeto della rappresentazione. Per ognuno di questi simboli si costruisce una funzione di appartenenza, secondo la definizione della teoria fuzzy. Si è scelto di fare ricorso a funzioni di tipo trapezoidali, come quelle della figura 5.2.
Per sfruttare al meglio la logica fuzzy queste funzioni sono tarate con l'andamento del profilo del gene in questione. A tal fine sono determinati i valori massimi e minimi del profilo del gene. Dopo ciò, sono disponibili diverse strategie implementative. L'approccio che si è deciso di utilizzare è quello di effettuare una scelta euristica, sull'andamento previsto del profilo. Nel caso, ad esempio, dei simboli di minimo e massimo, si è cercato di tradurre il comportamento di un agente umano, vale a dire, ad esempio, nel caso di un simbolo di massimo, questo è da considerare più significativo più si avvicina al valore massimo dell'intero

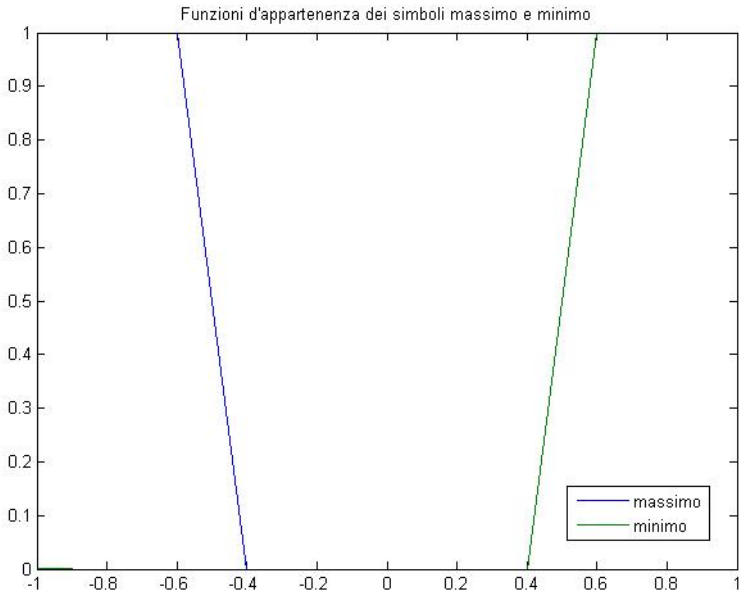


Figura 5.4: Esempio di funzioni di appartenenza campione per i simboli di minimo e massimo.

profilo. Si ha poi che, più ci si allontana da questo valore atteso, minore è la significatività del punto. Se si prende in esame, invece, il simbolo di minimo, il ragionamento da effettuare è l'esatto opposto.

A questo punto per ogni gene della rete che si vuole analizzare e per ogni simbolo dell'alfabeto, si è definita una funzione di appartenenza. Dopo di ciò, il procedimento continua, come nel caso dell'uso della rappresentazione simbolica, con la determinazione delle astrazioni temporali. Per ognuna di queste si utilizza il valore assoluto, del punto dominante in questione, per determinare un coefficiente fuzzy, tramite l'opportuna funzione di appartenenza.

Questi numeri fuzzy sono utilizzati nella fase successiva, che è molto simile a quella descritta precedentemente. Nell'estrazione della sottostringa o sottosequenza comune, si usa come termine di paragone, non solo la lunghezza della parte comune, ma un coefficiente che cerca di riassumere le seguenti caratteristiche attese: si ha una maggiore correlazione tra due profili genetici più lunga è la sottosequenza o sottostringa comune; un ulteriore contributo alla rassomiglianza è dato dall'omogeneità tra i valori fuzzy dei simboli. L'obiettivo è quello di verificare che sia mantenuto un rapporto di corrispondenza tra i simboli dei profili dei geni in entrambi i profili analizzati. In questo modo si riesce maggiormente ad individuare situazioni, che altrimenti sarebbero risultate problematiche, con un conseguente miglioramento delle prestazioni attese.

L'omogeneità tra i valori fuzzy, delle astrazioni simboliche, serve a codificare

la seguente ragionevole supposizione: se due geni sono tra di loro correlati e dalle porzioni dei loro profili fino ad ora analizzate, si ha una corrispondenza tra massimi, nella loro entità relativa, ci si attende che successivamente, trovando un minimo basso in un profilo, a questo corrisponda un minimo basso nell'altro. In questo senso è inteso questo contributo positivo, per l'omogeneità fra profili, alla misura di correlazione tra geni.
L'obiettivo atteso, nell'introduzione della logica fuzzy, è quello di mantenere la selettività, come accade utilizzando la rappresentazione simbolica, aumentando il numero di rapporti di correlazione correttamente reperiti. Come si vedrà meglio in seguito, questo corrisponde a voler aumentare il valore di richiamo dell'algoritmo complessivo, cercando di mantenere inalterato il valore di precisione, in modo da ottenere prestazioni ancora significative.

5.4 SVM (Support Vector Machine)

Le tecniche descritte fino a questo punto del capitolo sono state introdotte per la risoluzione del primo obiettivo della tesi, vale a dire il confronto tra l'uso di una rappresentazione numerica e una simbolica, in cui sono stati introdotti anche concetti della logica fuzzy, per il processo di determinazione della rete di regolazione genica.
In questo capitolo è introdotta la classe di algoritmi utilizzata per la risoluzione del secondo obiettivo, la ricerca di sottostrutture significative (FFL) all'interno della rete di regolazione. La classe di algoritmi utilizzata per questo scopo sono le SVM (Support Vector Machine). Come fattore comune tra i due obiettivi si è mantenuta la comparazione tra le due sopracitate rappresentazioni, come base di partenza per l'analisi dei dati di genomica.
Le SVM (Support Vector Machine) rappresentano la classe di algoritmi per la classificazione di dati che, al momento attuale, fornisce le prestazioni migliori. Questa grande efficienza, oltre che essere legata all'ambito di utilizzo, vale a dire le serie temporali dei profili genici, si può riassumere in due grandi vantaggi, che questi algoritmi forniscono:

1. Il processo di learning, che sta alla base delle SVM, è costruito su di un'idea semplice, ma al contempo efficace, che mostra in modo chiaro quali sono i passi che comunemente un agente umano utilizza nei processi reali di apprendimento;

2. Il modello essenziale delle SVM è sufficientemente complesso, nonostante la semplicità dell'idea che lo definisce; questo si nota anche dal fatto che questi algoritmi racchiudano al loro interno l'utilizzo molti altri algoritmi per l'apprendimento automatico, come ad esempio reti neurali, reti con funzioni a basi radiali (RBF) e appositi classificatori polinomiali.

L'intero processo di apprendimento può essere sintetizzato come un'operazione lineare di separazione, in un iperspazio, legato allo spazio di partenza tramite una funzione non lineare. Grazie allo sfruttamento di opportuni kernel è possibile effettuare tutte le operazioni computazionali direttamente nello spazio vettoriale di partenza, senza dover passare per l'iperspazio ausiliario.

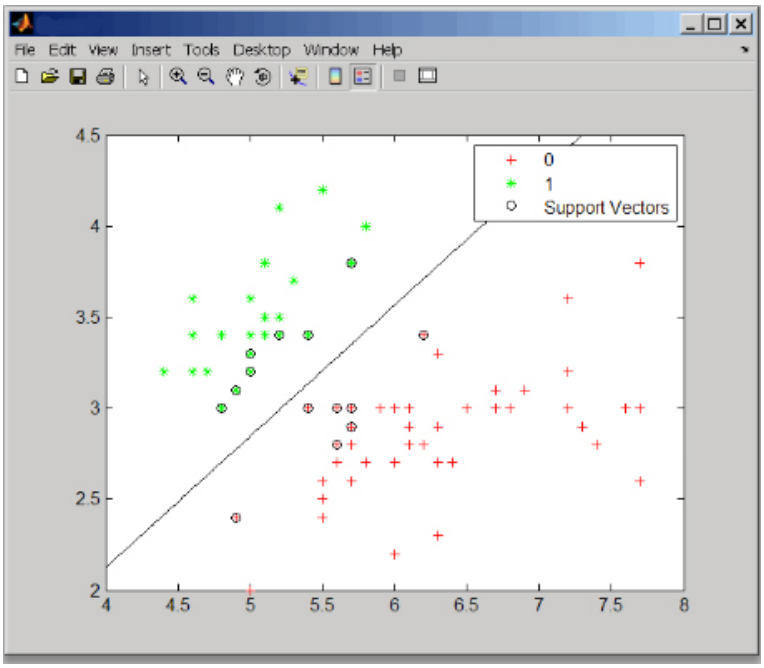


Figura 5.5: Esempio di un processo di classificazione tramite SVM di un dataset.

Il meccanismo di Machine Learning parte dalla fase di addestramento in cui la SVM riceve come input una collezione di vettori nella forma

$$c \ (x_1, y_1), \ldots, (x_n, y_n)$$

in cui le coppie (x_i, y_i) definiscono il pattern, costituito da n dimensioni, mentre il parametro c rappresenta la classe a cui questo pattern appartiene. Nell'ambito preso in esame in questa tesi, un generico pattern è costituito da delle coppie (x_i, y_i), in cui il vettore $\overline{x}$ rappresenta la sequenza progressiva degli istanti temporali, in cui i geni sono stati campionati ed il vettore $\overline{y}$ rappresenta la sequenza dei valori campionati.
La fase di training mira a determinare un'opportuna funzione, che mappa lo spazio delle features nelle rispettive classi di appartenenza. Per esigenze computative si richiede che questa funzione appartenga ad una determinata classe di funzioni. I classificatori delle SVM sono basati sulla classe degli iperpiani della forma

$$(w \cdot x) + b = 0$$

,

$$w \in \mathcal{R}^n$$

,

$$b \in \mathcal{R}$$

che definiscono le funzioni del tipo

$$f(x) = \text{sgn}(w \cdot x) + b$$

La scelta dell'iperpiano ottimo, definito come quello col massimo margine di separazione tra le due classi, che devono essere determinate, è univoca tramite la risoluzione di un problema di ottimizzazione di tipo quadratico, detto appunto QP. Questa è un'espansione di un sottoinsieme dei pattern di training. Questi, detti vettori di supporto, contengono tutta la porzione d'informazione rilevante, che riguarda il problema di classificazione e, dal punto di vista visivo, sono quelli che si trovano sul confine che separa le classi.

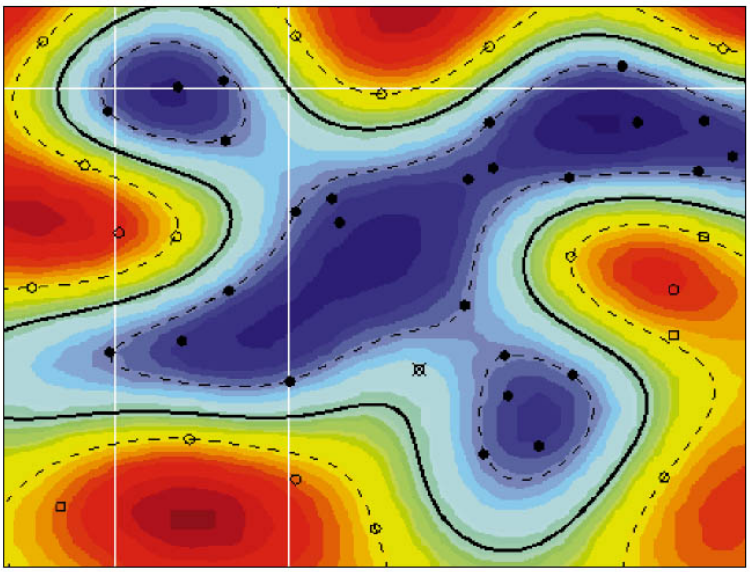


Figura 5.6: Esempio di classificatore mediante SVM.

L'idea di base delle SVM, vale a dire l'operazione di mapping dei dati di partenza in un opportuno spazio vettoriale, si basa su di una funzione di map, che ha la seguente definizione

$$\varphi : \mathcal{R}^n \rightarrow \mathcal{F}$$

Questa funzione rende lineare, in questo nuovo spazio vettoriale, l'operazione originale di separazione tra le due classi, che nello spazio di partenza poteva non esserlo. Dal punto di vista computazionale i calcoli principali consistono in una serie di prodotti vettoriali, per cui se si lavora in uno spazio vettoriale $\mathcal{F}$, che ha un numero elevato di dimensioni, il costo computazionale diventa sensibilmente elevato. Per questo motivo sono stati introdotti degli opportuni kernel k che permettono una valutazione più efficiente.
Esistono diverse tipologie di kernel. Quelli utilizzati in questo lavoro sono i seguenti: lineare, polinomiale e RBF. Il kernel polinomiale utilizzato per i nostri test ha la seguente forma

$$k(x, y) = (sxy + c)^d$$

I parametri che caratterizzano questo kernel sono i coefficienti $s \in \mathcal{R}$, $c \in \mathcal{R}$ e $d \in \mathcal{N}$.
Il kernel RBF utilizzato per i nostri test ha la seguente forma

$$k(x, y) = \exp\left(-\gamma \|x - y\|^2\right)$$

Il parametro che lo contraddistingue è il coefficiente $\gamma \in \mathcal{R}$. In generale, con quest'ultimo kernel, si ha che, usando elevati valori di γ le funzioni risultanti sono più flessibili, mentre per valori più piccoli, le funzioni sono più piatte, quindi meno selettive. Da questa considerazione ne segue che si preferisce scegliere valori elevati, nel caso si voglia riprodurre dei contorni di separazione più irregolari, mentre valori più piccoli, sono utilizzati nel caso in cui si voglia evitare situazioni di overfitting e per ignorare eventuali contributi dovuti ad errori statistici.
Riassumendo, gli aspetti principali che hanno portato all'utilizzo di questa classe di algoritmi sono i seguenti:

1. L'intero procedimento, dal punto di vista teorico, è basato completamente sulla teoria dell'apprendimento statistico;

2. Questi algoritmi sono efficienti, poiché permettono di ricondurre ad un problema di ottimizzazione quadratica, la cui soluzione può essere univocamente identificata;

3. Questo metodo contiene al suo interno una serie di algoritmi euristici, utilizzati a seconda della situazione, ad esempio in base alla scelta del kernel.

5.5 Valutazione del Processo di Reperimento dell'Informazione

Il processo di classificazione ha come obiettivo l'assegnazione di un insieme di pattern ad una determinata classe di appartenenza, nello specifico le classi uti-

lizzate segnalano o meno la presenza di un rapporto di correlazione tra geni e il segno di questo rapporto (positivo o negativo), nel caso del primo obiettivo della tesi e la tipologia di FFL, per il secondo obiettivo.
Il ramo della Computer Science, che si occupa della valutazione di questo processo di classificazione, è l'IR (Information Retrieval). Questo settore mette a disposizione una serie di metriche che permettono di avere una valutazione oggettiva dell'esito del processo di apprendimento.
Prima di cominciare ad elencare queste misure è necessario chiarire alcuni concetti base. Il processo di classificazione può portare essenzialmente a quattro differenti esiti:

1. Terne della classe positiva che sono classificate correttamente (denominate True Positive, o TP);
2. Terne della classe positiva che non sono classificate correttamente (denominate False Negative, o FN);
3. Terne della classe negativa che sono classificate correttamente (denominate True Negative, o TN);
4. Terne della classe negativa che non sono classificate correttamente (denominate False Positive, o FP).

Nel caso in esame, si ha che l'identificazione corretta di FFL o di geni correlati, porta maggiore informazione rispetto all'identificazione delle terne che non rappresentano strutture FFL o l'identificazione di geni non correlati. Il motivo, in parte, è dovuto al fatto che il classificatore esaminato, in entrambi i casi, lavora con dataset sbilanciati. In generale l'identificazione di FFL e di coppie di geni correlati è l'obiettivo principale dell'intero processo, per cui si parlerà di terne rilevanti reperite, in riferimento a questa situazione.
Per la misura della bontà di un classificatore si possono utilizzare diverse formule che forniscono delle metriche, permettendo così di paragonare tra loro algoritmi differenti; quelle analizzate nei test successivi sono le seguenti:

1. $Precisione = \frac{TP}{TP+FP}$, rappresenta la percentuale di terne rilevanti reperite in modo esatto, rispetto a tutte quelle estratte;
2. $Richiamo = \frac{TP}{TP+FN}$, rappresenta la percentuale di terne rilevanti reperite in modo esatto tra tutte quelle presenti;
3. $Accuratezza = \frac{TP+TN}{TP+TN+FP+FN}$, rappresenta la percentuale di terne correttamente classificate;
4. $F - measure = 2 * \frac{precisione*richiamo}{precisione+richiamo}$, misura che sintetizza i concetti inclusi nei valori di precisione e di richiamo;

5. $MCC = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$, (Matthew Coefficient Correlation) rappresenta un coefficiente di correlazione il cui valore è compreso tra -1 e 1.

In particolare si è fatto ricorso alla misura di correlazione MCC, in quanto il caso in esame è caratterizzato da una proprietà che rende alcune delle altre misure relativamente meno significative. Si ha, infatti, che le due classi da identificare, risultano sbilanciate. La misura di correlazione MCC fornisce un raffronto tra le prestazioni dell'algoritmo di classificazione e quelle di un classificatore casuale. Il coefficiente MCC assume valori compresi tra -1 e 1; più precisamente, vale uno in corrispondenza di una classificazione perfetta, in cui tutti i dati sono assegnati correttamente alla classe di appartenenza; assume valore -1 nel caso opposto; un valore pari a zero, invece, indica che il classificatore ha una prestazione paragonabile a quella ottenuta tramite scelta casuale della classe di appartenenza.

Capitolo 6

Ricostruzione della Rete di Regolazione

6.1 Introduzione

Questo capitolo della tesi si occupa di presentare i test condotti su dataset reali, per la ricostruzione della rete genica, il primo degli obiettivi che ci si era preposti. In questo settore, lo stato dell'arte è rappresentato dagli algoritmi ARACNE e REVEAL. Le prestazioni di questi sono dell'ordine del 40%, come precisione e richiamo nel processo di reperimento. Questo comportamento è il termine di paragone da utilizzare per la valutazione definitiva degli algoritmi fin qui proposti, ma manifesta, allo stesso tempo, una certa difficoltà intrinseca del problema, che ancora non è stata risolta.

Le prestazioni degli algoritmi attuali sono di per sè inferiori a quelle ottenute da un semplice classificatore casuale, nel quale la presenza di una dipendenza tra coppie di gene è decisa casualmente dal lancio di una moneta.

A causa di questa difficoltà, il problema generale può essere spezzato, oppure ricondotto a tutta una serie di sottoproblemi ad esso correlato. Uno di questi è, ad esempio, la ricerca di coppie di geni tra di loro in relazione, senza specificare quale dei due condiziona effettivamente l'altro. Grazie a questo procedimento è possibile determinare un primo insieme di grafi, a cui appartiene quello ricercato, restringendo così lo spazio di ricerca ed ottenendo una prima descrizione sommaria delle caratteristiche strutturali del grafo di regolazione globale.

Il passo successivo, del problema, è quello di scegliere il verso ed il tipo di regolazione. Bisogna, infatti, ricavare anche il tipo di relazione tra i geni, vale a dire se il gene condiziona positivamente o negativamente l'altro. Di quest'ultimo problema si sono occupati Sacchi [LS05] e Falda [Fal09] nei rispettivi lavori

di ricerca. Mentre Sacchi utilizza delle tecniche di Data Mining per ricavare il verso della regolazione, Falda ricorre alla rappresentazione simbolica, descritta in precedenza, per lo stesso scopo. Entrambi i metodi producono, su dati reali, un'accuratezza elevata, dell'ordine del 90%.
Entrambi partono dall'assunzione dell'esistenza della dipendenza, per ricavarne il verso effettivo, per cui l'effettivo utilizzo di tali metodi, nel problema generale di ricostruzione dell'intera rete, dipende pesantemente dal passo precedente, in termini di efficenza.
In questo capitolo ci si occuperà di paragonare tra loro i seguenti, metodi per la determinazione della correlazione tra geni:

1. Correlazione, ricavata dalla distanza euclidea tra i profili genici;
2. DTW (Dynamic Time Warping), descritto nell'appendice (A.4);
3. LCS-CAL (Longest Common Sequence CAL);
4. LCString-CAL (Longest Common Substring CAL).

Gli ultimi due metodi sono stati descritti in dettaglio nel capitolo precedente (5.2). Una suddivisione della fase di testing, è stata effettuata sulla rappresentazione utilizzata come base per la metrica di correlazione. Saranno confrontate tra loro le prestazioni ottenute usando i profili numerici dei geni, rispetto a quelle risultanti dall'utilizzo delle descrizioni simboliche.
Come alfabeto della rappresentazione simbolica si sono utilizzati i seguenti simboli: massimo (M), minimo (m), crescente (c), decrescente (d), flesso (f), stazionario (s), zero (z), saturazione(t). Nella pratica si è notato che il simbolo di flesso risulta, delle volte, troppo discriminante, nel processo di estrazione della sottosequenza o sottostringa comune. Per questa ragione si è deciso di suddividere ulteriormente i test, in base all'utilizzo o meno di tale simbolo, nelle tecniche LCString e LCS.
Sono stati utilizzati tre dataset. Il primo proviene da esperimenti reali su di un sottoinsieme di geni che partecipano al ciclo cellulare del lievito. I rimanenti due dataset, sono stati generati artificialmente, nell'ambito del progetto, sfida, denominato dream3 e si differenziano per la presenza o meno di un fattore di errore, inserito nel processo di creazione degli andamenti genici. Per la valutazione dell'intero processo di classificazione saranno utilizzate le metriche, derivate dalla teoria del reperimento dell'informazione.

6.2 Confronto tra Metodi

I dataset utilizzati per questa fase di test appartengono essenzialmente a due esperimenti. Il primo dataset si riferisce ad un esperimento reale, sul ciclo cellulare del lievito. Questo è costituito da un'unica rete, per la quale sono presenti

quattro distinte repliche dell'esperimento, vale a dire quattro set di dati. La rete di regolazione da ricavare è costituita da 24 geni, i cui profili sono campionati in 14 punti consecutivi.
L'altro dataset, denominato dream3, dal nome del progetto, sfida che l'ha prodotto, è costituito da cinque reti, per ognuna delle quali sono presenti quattro set distinti, tutti formati da 10 geni, campionati in 21 punti. In questo caso i dati sono ottenuti da una simulazione sperimentale. Per questo dataset sono presenti due versioni distinte. Queste si differenziano per l'aggiunta, nella fase di simulazione, di una componente casuale d'errore. Il test degli algoritmi fin qui elencati, avverrà sia sulla versione non rumorosa, sia su quella rumorosa. Per quest'ultima versione si verificherà il comportamento degli algoritmi, in concomitanza dell'utilizzo o meno di tecniche di filtraggio come fase di pre-analisi.

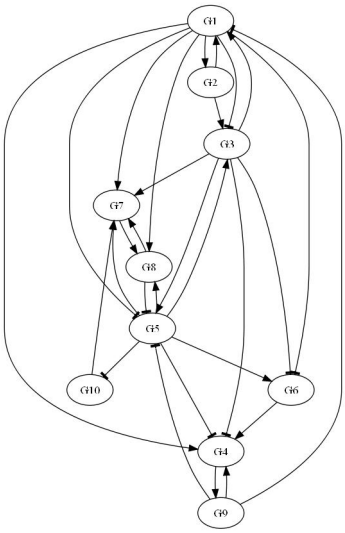
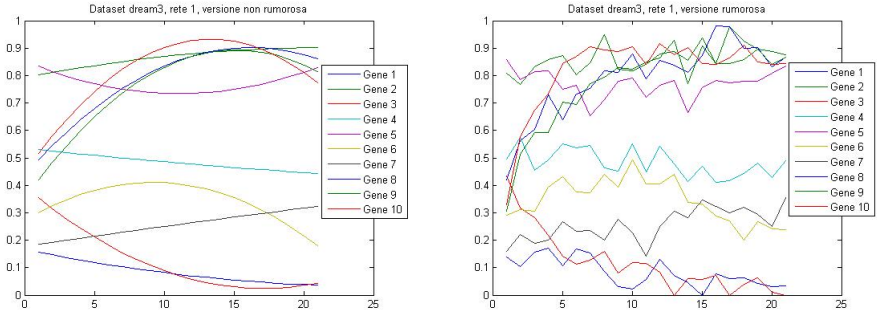


Figura 6.1: Esempio di rete di regolazione genica da ricostruire.

Per quanto riguarda i test con gli algoritmi LCString-CAL e LCS-CAL (d'ora in poi denominati semplicemente LCString e LCS, per semplicità), si effettuerà una distinzione, sull'alfabeto utilizzato per la rappresentazione simbolica: la discriminazione si avrà con l'utilizzo o meno del simbolo di flesso nella rappresentazione simbolica. Il motivo, come precedentemente accennato, è il suo elevato apporto discriminatorio. In pratica si ha che, sperimentalmente, questo simbolo è molto sensibile all'andameno del profilo genico, per cui delle piccole alterazioni di tali profili, porta all'assegnamento o meno di alcuni punti dominanti a questo simbolo. L'altro obiettivo, di questi test, è quello di confrontare tra loro metodi basati sulla rappresentazione numerica, come il calcolo della correlazione e della DTW, e metodi basati sulla rappresentazione simbolica, gli altri algoritmi. Si valuteranno, infine, le modifiche agli algoritmi LCString e LCS, tramite l'introduzione dei concetti della logica fuzzy.



(a) (b)

Figura 6.2: Esempi di andamenti dei profili genici, dataset dream3, versione non rumorosa (figura (a)) e versione rumorosa (figura (b)).

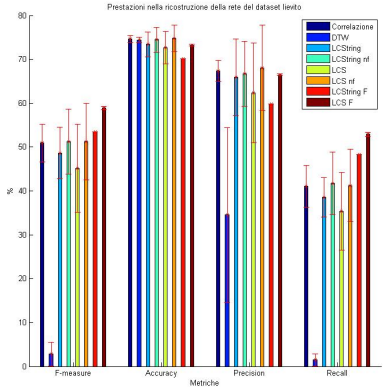


Figura 6.3: Prestazioni degli algoritmi sul dataset del lievito.

Nella figura 6.3 sono riportati i risultati medi, ottenuti sui vari set, testando i metodi precedentemente elencati, con la corrispettiva varianza. Per avere un quadro completo della situazione bisogna osservare che, come spesso accade, in questo ambito di analisi, le reti di regolazione genica sono contraddistinte da matrici molto sparse, per cui un primo classificatore che è possibile realizzare e che funge bene da paragone è il cosiddetto classificatore di maggioranza.
Questo semplice scelta permette di ottenere, nel processo di reperimento delle correlazioni tra geni, un classificatore con prestazioni costanti. Nel caso in esame, dal dataset del lievito è possibile ricavare che le coppie di geni tra loro correlati sono solo il 20% di tutte le coppie possibili. Con questa configurazione il classificatore ottiene un'accuratezza dell'80%, assegnando tutte le coppie di geni alla classe dei geni non correlati.
Osservando i risultati ottenuti, si nota che questa accuratezza non viene raggiunta da nessun metodo. Dato il contesto di analisi, però, conviene analizzare tutte le

misure ricavate dai test, in modo da avere una descrizione più completa dell'intero processo. In particolare è possibile considerare la F-measure, che racchiude i concetti relativi alle misure di precisione e richiamo.
Si nota subito che utilizzando la DTW si ottengono prestazioni sempre peggiori, rispetto agli altri casi, in quanto questo metodo non è riuscito a trovare praticamente nessuna correlazione tra geni. Per quanto riguarda invece le altre strategie, le prestazioni sono pressappoco dello stesso ordine di grandezza. Il metodo tramite correlazione, che utilizza la rappresentazione numerica, ha prestazioni leggermente peggiori degli altri algoritmi, mentre le varianti fuzzy dei metodi LCString e LCS risultano invece leggermente migliori.
Nelle figure 6.5, 6.6 e 6.7 sono riportati i risultati ottenuti con i dataset dream3. Nella figura 6.5 sono riportati i risultati medi per le cinque reti del dataset prese in esame. In questo caso il dataset è privo della componente d'errore nella fase di generazione dei dati. Nella figura 6.6 i risultati si basano sulla versione rumorosa dei dati, utilizzati direttamente dai vari algoritmi. Nella figura 6.7, invece, si è fatto ricorso ad una fase di filtraggio preliminare, per cercare di eliminare l'errore statistico presente nei dati.
Nell'immagine 6.4 sono riportati i risultati medi complessivi, al variare delle versioni del dataset dream3.

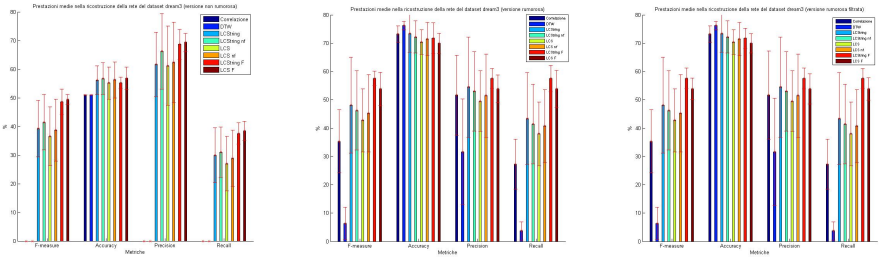


Figura 6.4: Grafici medi delle prestazioni degli algoritmi sul dataset dream3.

Come accadeva nel caso precedente, i metodi basati sulla rappresentazione numerica dei profili dei geni, risultano staticamente peggiori, rispetto a quelli basati sulla rappresentazione simbolica. In questi dataset, la considerazione sul classificatore di maggioranza, come metro di paragone, vale solamente per le prime tre reti del dataset. Nelle restanti reti, la matrice di correlazione non risulta molto sparsa, per cui questo classificatore perde di significato, nel suo scopo di paragone.
Allo stesso modo del dataset del lievito, risulta che le varianti fuzzy dei metodi LCString e LCS forniscono dati leggermente migliori.

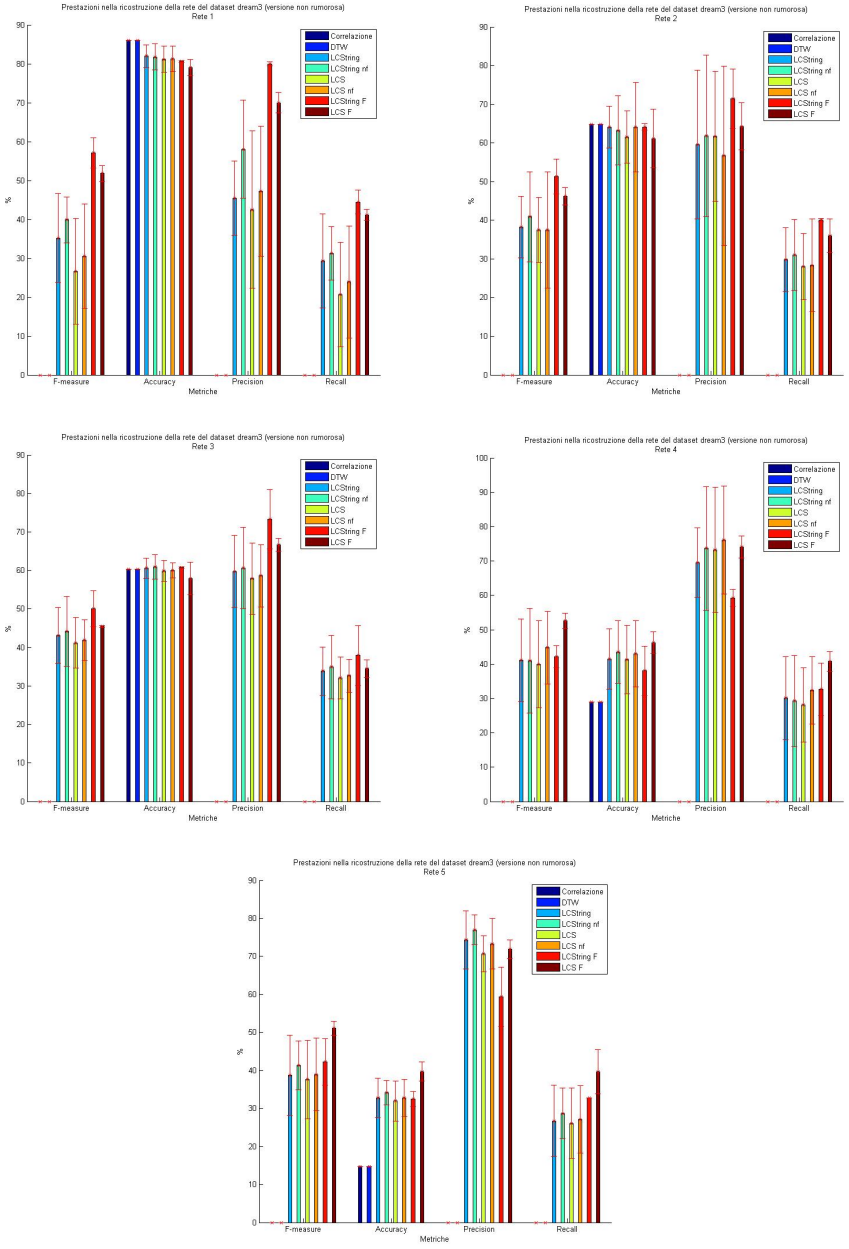


Figura 6.5: Prestazioni degli algoritmi sul dataset dream3, versione non rumorosa.

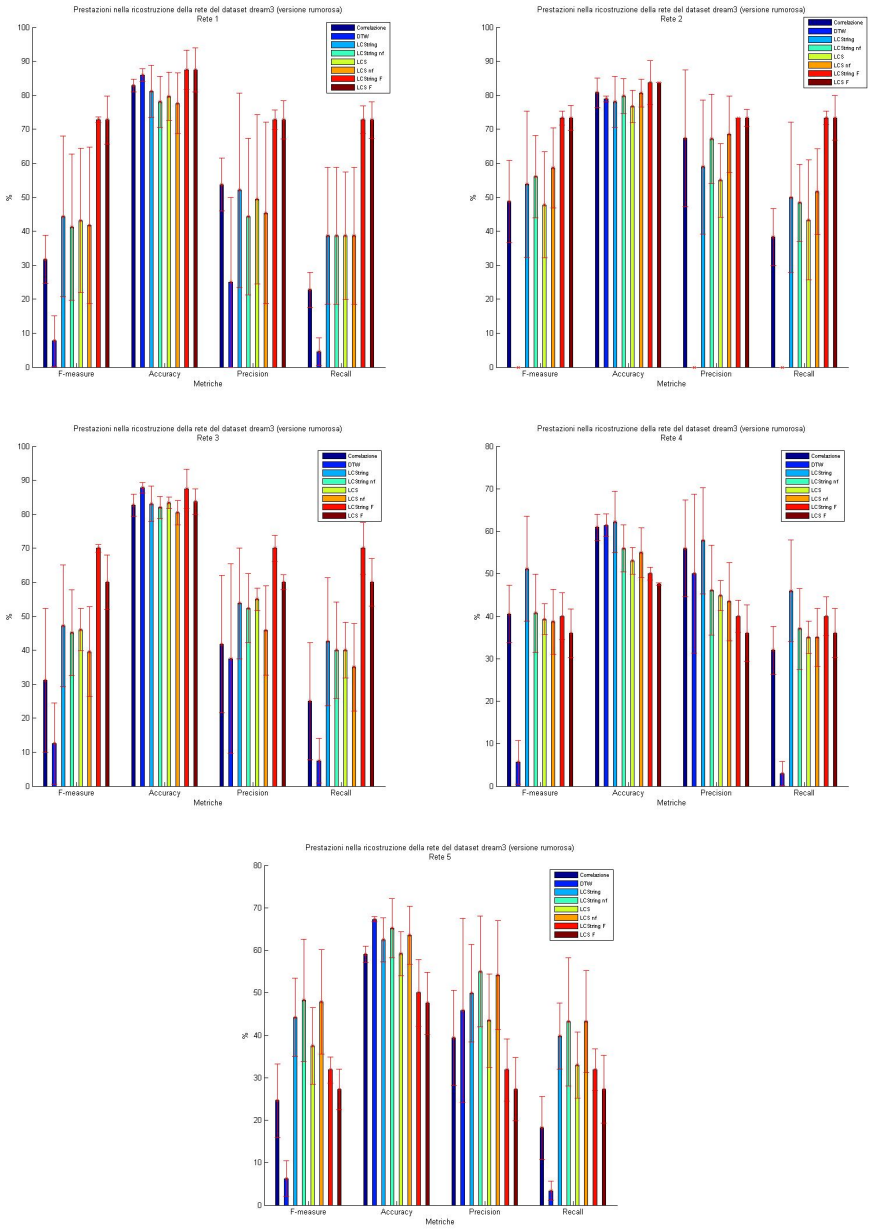


Figura 6.6: Prestazioni degli algoritmi sul dataset dream3, versione rumorosa non filtrata.

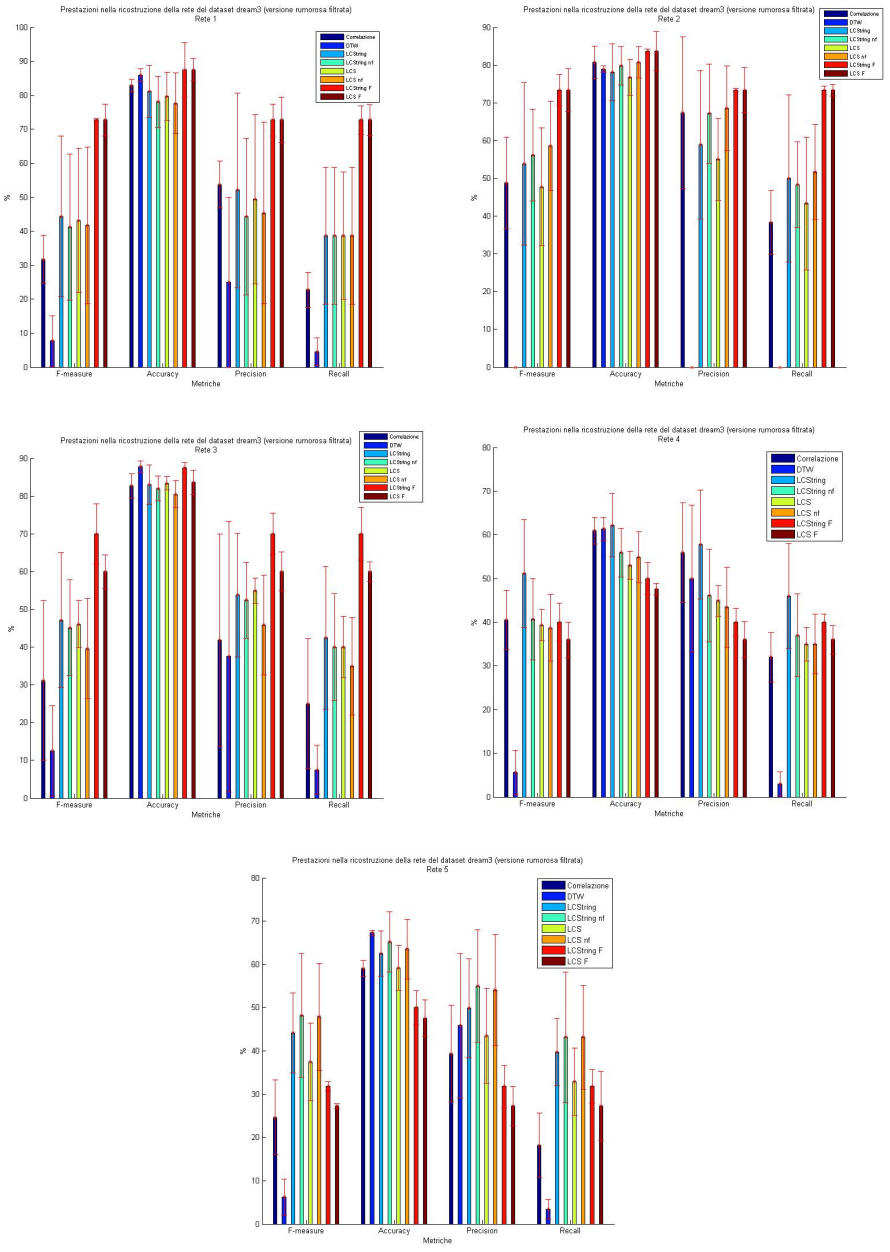


Figura 6.7: Prestazioni degli algoritmi sul dataset dream3, versione rumorosa filtrata.

6.3 Reperimento di Relazioni Complesse

I ragionamenti mostrati fino a questo punto si occupano del reperimento di relazioni solo tra coppie di geni. Questo meccanismo può essere esteso ad un insieme

di k geni. In questo capitolo ci si occuperà di trattare il caso specifico di $k = 3$, vale a dire considerare terne di geni.

Per descrivere quest'analisi si utilizzeranno tre profili generati artificialmente. I tre geni, campionati in 11 punti, sono denominati rispettivamente $C2$, $C4$ e $C6$ e sono mostrati in figura 6.8. Nei casi base, analizzati in precedenza, il calcolo della misura di correlazione avveniva tra coppie di geni. In questa nuova situazione è necessario considerare anche il caso aggiuntivo per cui siano due i geni che determinano l'andamento di un altro gene.

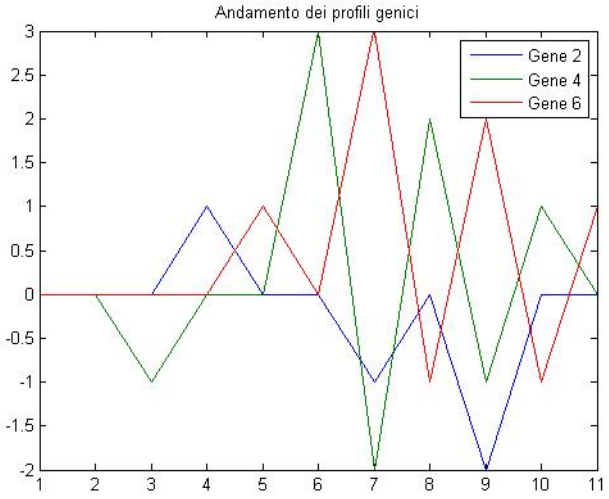


Figura 6.8: Profili di tre geni generati artificialmente.

Nel calcolo delle misure di correlazioni si ha la necessità, che non si verificava nei casi precedenti, d'introdurre una strategia per la combinazione di più profili genici. Uri Alon, nel suo lavoro sulle reti di regolazione [Alo07], introduce alcune funzioni logiche per questo scopo. Quelle utilizzate in questo esempio sono le seguenti: $C2 \wedge C4$, $C2 \vee C4$, $C2 \wedge \overline{C4}$ e $\overline{C2} \wedge C4$.

Nel caso preso in esame risulta che il gene $C6$ è condizionato positivamente dalla combinazione dei geni $C2$ e $C4$, secondo la funzione $C2 \vee C4$.

Questo metodo di operare ha un problema dal punto di vista computazionale: si ha, infatti, che analizzando terne gli algoritmi per il calcolo delle LCS o delle LCString, sono eseguiti un numero cubico di volte, rispetto al numero di geni presenti nella rete analizzata. Questo risulta gravoso, nel caso in cui il numero di geni da analizzare non sia sufficientemente piccolo. Nonostante ciò, questi passaggi rappresentano lo stato attuale dell'arte, nel caso generale delle dipendenze tra insiemi di geni.

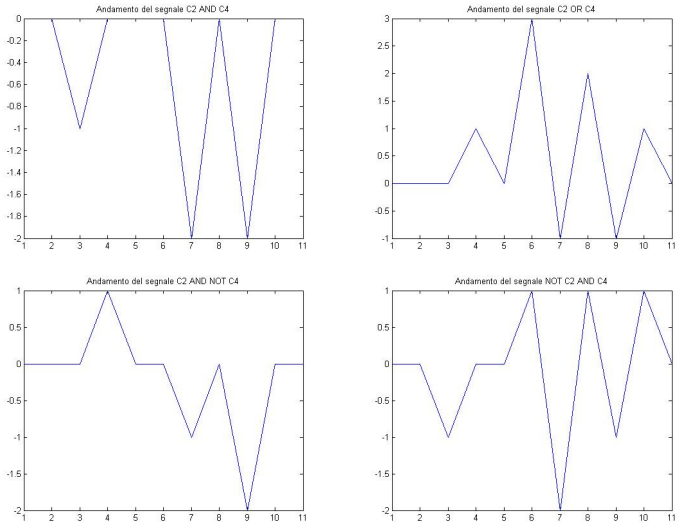


Figura 6.9: Combinazioni possibili tra andamenti genetici.

6.4 Conclusioni

Nella comparazione dei metodi, basati sulla rappresentazione numerica, con quelli basati sulla rappresentazione simbolica, si è riconosciuto univocamente che i primi forniscono sempre prestazioni peggiori, rispetto ai successivi.
Comparando più nello specifico i metodi LCString e LCS, si osserva che le prestazioni sono tra loro molto simili, anche se i metodi LCString risultano mediamente migliori, anche se di poco. Un'altra considerazione ottenuta è che con l'esclusione dei flessi, nella descrizione simbolica dei geni, si ha un leggero miglioramento nelle prestazioni, sia con il metodo LCString che con il metodo LCS.
Un'altra importante considerazione ottenuta è che le modifiche fuzzy portano ad un ulteriore miglioramento, con un aumento dei valori di richiamo, mantenendo i valori di precisione sostanzialmente inalterati.
Come possibile sviluppo, di questa classe di algoritmi, si è individuata la possibilità di adattare le membership function, tramite algoritmi di apprendimento automatico, sugli andamenti dei geni. Il motivo risiede nel fatto che, nell'implementazione utilizzata per questi test, questo adattamento è fissato in maniera statica sugli andamenti, mentre con una procedura di apprendimento automatico, sarebbe possibile caratterizzare maggiormente queste funzioni, in modo da migliorare le prestazioni, utilizzando una porzione di dati come training dell'algoritmo.
Un'ulteriore possibilità di miglioramento è l'utilizzo di una classe differente di funzioni come membership function, utilizzando, ad esempio, delle distribuzioni normali, al posto delle funzioni trapezoidali, usate per questi test.
Nonostante le buone prestazioni ottenute, si ha che queste non rappresentano

ancora un miglioramento dello stato dell'arte, per questo settore d'analisi, ma si avvicinano molto ad esse.

Capitolo 7

Reperimento delle Strutture FFL

7.1 Introduzione

Nell'arco degli ultimi cinquanta anni lo studio delle tecniche di Machine Learning si è sviluppato notevolmente, grazie soprattutto al grande contributo derivante dall'aumento delle capacità di calcolo degli elaboratori. Com'è accaduto per gli algoritmi, che stanno alla base di questa disciplina, anche gli obiettivi sono mutati nel tempo: si è spaziato da software per l'emulazione dell'intelligenza umana, in ambito ludico, ad analisi di tipo statistico di grandi moli di dati, come ad esempio accade con le procedure di data mining, negli ambiti di basket analisis o del rilevamento delle frodi.
L'obiettivo principale delle tecniche di Machine Learning è rispondere alla seguente questione

"Com'è possibile programmare computer in grado di imparare dall'esperienza, e quali sono le principali regole che definiscono il processo di apprendimento".

Gli ambiti d'impiego di questa disciplina sono numerosi, come ad esempio la programmazione di automi, in grado di muoversi ed imparare dalla propria esperienza la struttura del luogo in cui si trovano, oppure come estrarre, tramite data mining, dell'informazione significativa, contenuta all'interno di dati relativi a pazienti affetti da una malattia, in modo da poter conoscere a priori quale sarà il trattamento che, statisticamente, otterrà i migliori risultati su di un paziente futuro.
L'intero procedimento, che sta alla base di questo settore, può essere sintetizzato con i seguenti elementi che lo costituiscono. In corrispondenza di un preciso obiettivo il processo di Machine Learning costruisce un'opportuna metrica e memorizza la propria esperienza, costituendo un proprio modello interno. Questi

passi racchiudono in sé due grosse problematiche: la prima, che riguarda l'ambito della Computer Science, risponde alla domanda *"Come possiamo programmare computer che risolvano problemi e quali altri problemi possono essere trattati in maniera equivalente"*; la seconda, invece, riguarda l'ambito della Statistica e risponde alla domanda *"Cosa è possibile inferire da un ampio set di dati, a partire da un insieme di assunzioni iniziali"*. Il Machine Learning comprende entrambe queste problematiche, ma le riformula con un approccio che punta all'automazione. La prima questione è rivista in chiave della programmazione, che è considerata come finalizzata alla ricerca dell'autodeterminazione di algoritmi per la risoluzione dei problemi. La seconda, invece, è estesa alla ricerca di quali algoritmi risolvono efficientemente un determinato problema e come è possibile costituire più livelli d'inferenza d'informazione significativa e di come questi livelli possano essere combinati tra loro.
Rientrano nel processo di Machine Learning anche molte tematiche relative alla Psicologia. In particolare, svolge un ruolo importante la definizione dei concetti collegati al processo di apprendimento umano. Questo contributo, al momento attuale, è ancora marginale, se riferito a quello derivante dalla Computer Science o dalla Statistica. La ragione risiede principalmente, nella difficoltà che si ha nel formalizzare i passi che descrivono il processo di apprendimento umano oppure animale. Nonostante ciò, coll'aumentare della sinergia tra tutte queste discipline, si stanno sviluppando numerose strategie, che prendono spunto da osservazioni della natura, per definire nuove strategie e nuovi algoritmi per l'apprendimento automatico.
Tra le varie tematiche, tipiche del processo di Machine Learning, questo lavoro si è prefisso lo scopo di testare e valutare le prestazioni di un algoritmo di classificazione. L'ambito è l'inferenza di regole associative tra geni appartenenti a delle fissate reti di regolazione genetica. Nello specifico si è effettuato un test tramite un classificatore SVM, per inferire particolari sottostrutture significative, presenti all'interno di alcune reti genetiche, denominate FFL (Feed Forward Loop).

7.2 Determinazione dei Parametri Ottimi dell'Algoritmo SVM

Per la valutazione di un classificatore tramite algoritmo SVM si è utilizzata l'implementazione SVM^{light} [Joa08a], realizzata da Thorsten Joachims, docente del dipartimento di Computer Science della Cornell University. Il linguaggio di programmazione utilizzato è il c e la versione è la 6.02 del 14.08.2008. Un altro software a cui si è fatto ricorso è SVMsequel [III04], un'implementazione realizzata da Hal Daume III, professore del dipartimento di Computer Science dell'Università dello Utah, differente dell'algoritmo SVM^{light}, in quanto permette anche l'utilizzo di kernel per l'elaborazione diretta di stringhe.

Come dati per l'analisi si è utilizzato il dataset dream3, la versione non rumorosa, precedentemente utilizzata, che ricordiamo ha le seguenti caratteristiche:

1. Costituito da cinque reti;
2. Ogni rete ha a disposizione quattro differenti set di dati;
3. Ogni rete è composta da dieci geni;
4. Ogni gene è descritto da una sequenza formata da ventuno campioni.

L'obiettivo prefissato è stato testare la capacità del classificatore SVM, d'identificare delle sottostrutture significative, costituite da tre geni, presenti nelle reti di regolazione genetica. Nello specifico si è cercato di identificare tutti i FFL, presenti nelle reti prese in esame. Prima di procedere con l'apprendimento automatico, tutte le istanze dei FFL sono state effettivamente determinate. Come si osserva dalla tabella riassuntiva 7.1, solo tre reti presentano almeno una tipologia di FFL.

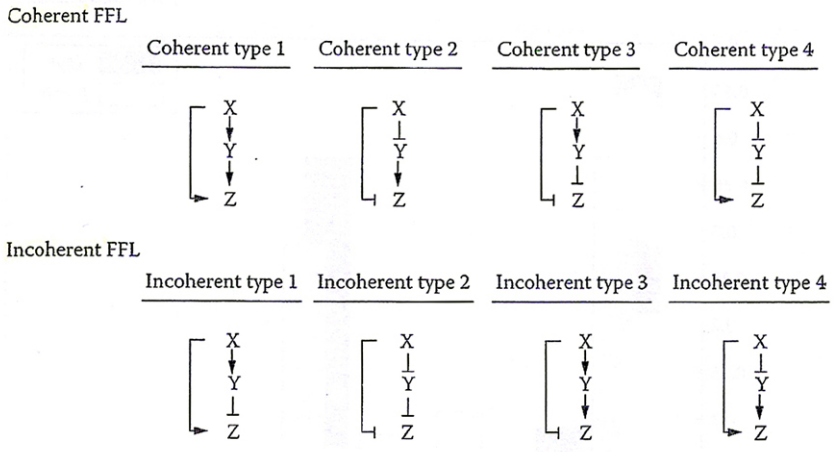


Figura 7.1: Tipologie di FFL (Feed Forward Loop).

	FFL coerenti				FFL incoerenti			
Rete	**1° tipo**	**2° tipo**	**3° tipo**	**4° tipo**	**1° tipo**	**2° tipo**	**3° tipo**	**4° tipo**
3	1	0	3	1	2	0	1	0
4	3	0	2	0	0	1	3	0
5	2	1	5	1	0	0	3	0

Tabella 7.1: Istanze di FFL presenti, nelle reti prese in esame.

Data la struttura dei dati analizzati, si è scelta una strategia iniziale di 3-cross validation, per l'ottimizzazione dei parametri del classificatore, che sarà poi testata su di altri dati. Per identificare i dataset utilizzati, si farà ricorso alla seguente notazione:

1. $\{S_i, \quad i = 1, 2, 3, 4\}$; il dataset S_i rappresenta l'insieme dei dati relativi ai set i relativi alle tre reti prese in esame.

Dato che i FFL sono strutture composte da tre geni, il singolo pattern utilizzato dal classificatore avrà la seguente forma

$$c\,(t_1, x_1) \ldots (t_n, x_n)\,(t_1, y_1) \ldots (t_n, y_n)\,(t_1, z_1) \ldots (t_n, z_n)$$

I vettori $\overline{x}$, $\overline{y}$ e $\overline{z}$ sono le rappresentazioni dei tre profili dei geni, mentre il vettore $\overline{t}$ è la sequenza degli istanti di campionamento.
Per la valutazione si è partiti da un insieme contenuto di terne, identificate tramite il parametro c, secondo la seguente simbologia: i pattern, appartenenti alla classe positiva, sono quelli in cui la terna di geni corrisponde ad un'effettiva struttura FFL, presente nella rete; quelli invece che appartengono alla classe negativa, sono quelli in cui la terna di geni non corrisponde ad una struttura FFL presente nella rete. Le sequenze dei profili sono sensibili all'ordinamento, dei geni che compongono la terna, grazie all'utilizzo della codifica mostrata in figura 7.1. Ad esempio, grazie a questa proprietà, i seguenti sei pattern

$$c_1(\overline{t}, \overline{x})(\overline{t}, \overline{y})(\overline{t}, \overline{z});\ c_2\,(\overline{t}, \overline{x})\,(\overline{t}, \overline{z})\,(\overline{t}, \overline{y})\,;\ c_3\,(\overline{t}, \overline{y})\,(\overline{t}, \overline{x})\,(\overline{t}, \overline{z})\,;$$
$$c_4(\overline{t}, \overline{y})(\overline{t}, \overline{z})(\overline{t}, \overline{x});\ c_5(\overline{t}, \overline{z})(\overline{t}, \overline{x})(\overline{t}, \overline{y});\ c_6(\overline{t}, \overline{z})(\overline{t}, \overline{y})(\overline{t}, \overline{x}).$$

sono tra di loro distinti, perché rappresentano delle differenti strutture, una volta fissata la tipologia di FFL analizzata.
Tra tutte le tipologie di FFL si è scelto d'identificare i FFL del primo, del terzo e del quarto tipo, per quanto riguarda quelli coerenti e del primo e del terzo tipo, per quelli incoerenti. La ragione è facilmente deducibile dalla tabella 7.2, dato che sono queste le tipologie che presentano una numerosità sufficientemente significativa d'istanze presenti. Ritornando ai pattern mostrati in precedenza, si nota che, una volta fissata la tipologia di FFL, che il classificatore cercherà di identificare, l'ordine in cui si presentano i profili dei geni, specifica univocamente le relazioni presenti tra i geni che costituiscono la terna. Nello specifico i sei pattern rappresentano tutte le possibili permutazioni dei tre geni x, y e z.
Come prima fase della valutazione del classificatore, sono state identificate tutte le istanze delle terne, che costituiscono una determinata tipologia di FFL, presenti nelle reti analizzate. Queste sono state classificate come appartenenti alla classe positiva. Per ognuna delle terne appena trovate, si sfrutta la significatività dell'ordinamento dei geni per costituire le istanze della classe negativa. Partendo da una terna della classe positiva si usano tutte le possibili permutazioni dei tre geni in questione, per ottenere dei nuovi pattern, che saranno assegnati appunto alla classe negativa.
A questo punto si hanno a disposizione dei dataset sbilanciati, in cui $\frac{1}{6}$ dei pattern appartengono alla classe positiva e i rimanenti $\frac{5}{6}$ appartengono alla classe negativa.

7.3 Valutazione del Classificatore SVM Monoclasse

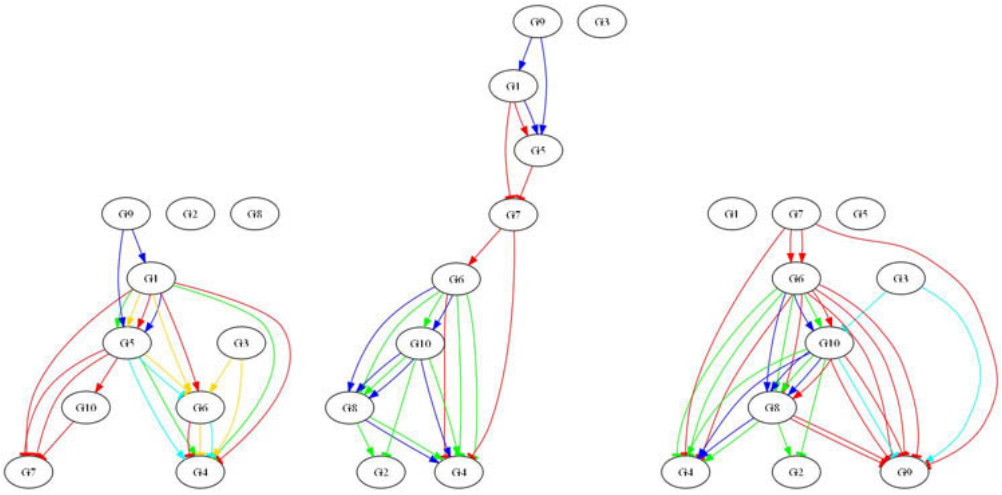
L'obiettivo preso in considerazione in questo capitolo è valutare il comportamento di un classificatore tramite SVM, per la determinazione di una singola tipologia di FFL. Tra tutte i FFL sono stati analizzati solo quelli che, statisticamente, nel dataset preso in esame, presentano una numerosità sufficientemente significativa, vale a dire i FFL coerenti del primo, del terzo e del quarto tipo e i FFL incoerenti del primo e del terzo tipo.
I dati di partenza sono stati suddivisi in 4 dataset indicati secondo la notazione:

- $\{S_i, \quad i = 1, 2, 3, 4\}$, in cui il dataset S_i rappresenta l'insieme dei dati relativi ai corrispettivi set i delle tre reti prese in esame.

Un'importante suddivisione dei test si ha in base al tipo di rappresentazione dei profili genetici utilizzata:

1. Rappresentazione numerica, si utilizzano i valori numerici campionati;
2. Rappresentazione simbolica, si utilizza la rappresentazione in simboli (vista nel capitolo 5.2).

Utilizzando queste rappresentazioni ne deriva che nel primo caso il singolo pattern, costituito dalla concatenazione dei tre profili numerici, ha una lunghezza di 63 features (il singolo gene è descritto da 21 campioni). Nella seconda, invece, ha una lunghezza decisamente inferiore e in generale si ha che la descrizione simbolica di un gene ha un numero di simboli differenti rispetto a quella di un altro. Nel caso del dataset dream3, gli andamenti di tutti i geni sono molto regolari, per cui il numero di simboli, a fronte dei 21 campioni di partenza, è di poche unità. Per questa ragione, come si vedrà maggiormente in seguito, il comportamento del classificatore, basato su rappresentazione simbolica, non da luogo a prestazioni significative.
La notazione utilizzata nella figura 7.2 è la seguente: le frecce indicano una regolazione positiva, le barre indicano, invece, una regolazione negativa; gli archi blu sono i FFL coerenti del primo tipo, gli archi rossi sono i FFL coerenti del terzo tipo, gli archi azzurri sono i FFL coerenti del quarto tipo, gli archi gialli sono i FFL incoerenti del primo tipo ed infine gli archi verdi sono i FFL incoerenti del terzo tipo.
Tutte le terne rappresentate in quest'immagine, sono tutti i pattern appartenenti alla classe positiva dei dataset, che saranno utilizzati dal classificatore. Le istanze della classe negativa, vale a dire le terne che non corrispondono ad una tipologia di FFL che l'algoritmo SVM deve classificare, sono ottenute, secondo l'idea precedentemente descritta, tramite permutazioni.
I quattro dataset di partenza sono stati suddivisi in due insiemi, uno per la fase di training e uno per la fase di testing. La fase di training è stata effettuata



(a) (b) (c)

Figura 7.2: Istanze FFL presenti nei grafi presi in esame(figura (a) rete 3, figura (b) rete 4, figura (c) rete 5).

sui dataset S_1, S_2 e S_3, mentre quella di testing sul dataset S_4. Per il training si è scelto di utilizzare una 3-cross validation, con ottimizzazione manuale dei parametri dei kernel. In seguito a questa fase inziale, si è scelta la configurazione ottimale per i tre rispettivi kernel utilizzati (lineare, polinomiale e RBF) e la si è utilizzata per la fase di testing.
La procedura di 3-cross validation permette di utilizzare, a rotazione, due dei tre set scelti per la validazione del set rimanente. In pratica si effettuano tre prove per ogni kernel da validare, secondo lo schema riportato in figura 7.3.

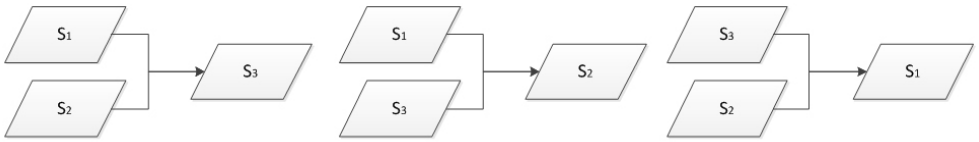


Figura 7.3: Passi della procedura di 3-cross validation sui tre dataset S_1, S_2 e S_3.

Per ognuna delle tre prove sono ricavate le misure di accuratezza e di MCC, in base al variare dei parametri interni dei kernel. Per ognuna di queste configurazioni si ricavano i valori medi nelle tre prove, in modo da determinare la configurazione ottimale. Di seguito sono riportate le prestazioni ottenute con questa procedura sul dataset dream3.

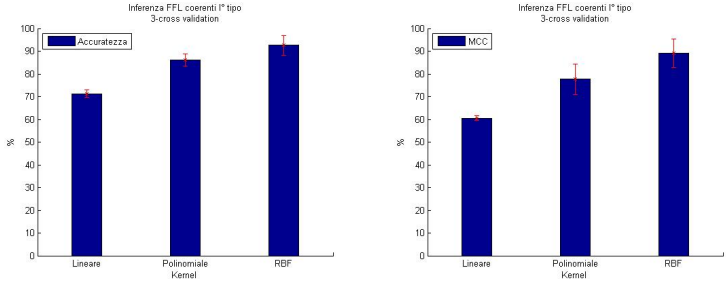


Figura 7.4: Risultati 3-cross validation dei FFL coerenti del primo tipo.

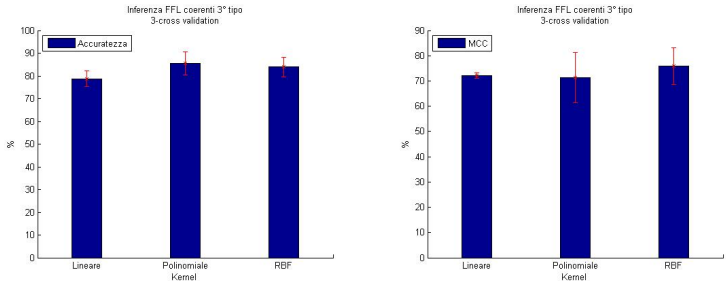


Figura 7.5: Risultati della 3-cross validation dei FFL coerenti del terzo tipo.

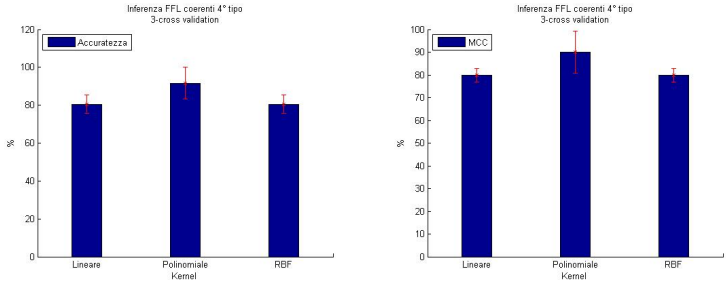


Figura 7.6: Risultati della 3-cross validation dei FFL coerenti del quarto tipo.

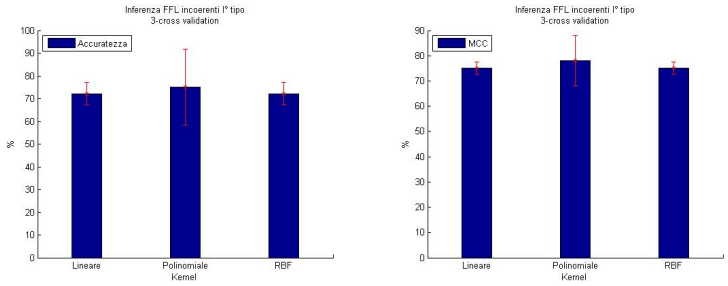


Figura 7.7: Risultati della 3-cross validation dei FFL incoerenti del primo tipo.

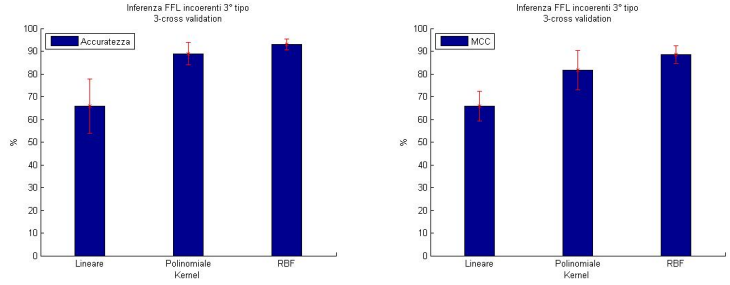


Figura 7.8: Risultati della 3-cross validation dei FFL incoerenti del terzo tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	71,296	0,211	60,542
Polinomiale	86,111	0,553	77,664
RBF	92,593	0,781	89,037

Tabella 7.2: Risultati 3-cross validation dei FFL coerenti del primo tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	77,778	0,443	72,157
Polinomiale	85,556	0,427	71,340
RBF	83,889	0,517	75,874

Tabella 7.3: Risultati 3-cross validation dei FFL coerenti del terzo tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	80,556	0,598	79,901
Polinomiale	91,667	0,802	90,117
RBF	80,556	0,598	79,901

Tabella 7.4: Risultati 3-cross validation dei FFL coerenti del quarto tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	72,222	0,502	75,092
Polinomiale	75	0,561	78,028
RBF	72,222	0,502	75,092

Tabella 7.5: Risultati 3-cross validation dei FFL incoerenti del primo tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	65,873	0,316	65,783
Polinomiale	88,889	0,631	81,563
RBF	92,857	0,772	88,604

Tabella 7.6: Risultati 3-cross validation dei FFL incoerenti del terzo tipo.

Per giudicare al meglio i risultati ottenuti, è necessario fare una riflessione preliminare sulla struttura del dataset, utilizzato nel processo di classificazione. Questo è strutturato in due classi: la classe positiva è meno numerosa di quella negativa. Analizzando più in dettaglio si osserva che per ogni istanza della classe positiva, ci sono cinque istanze della classe negativa. Il motivo risiede nella tecnica utilizzata per costruire il dataset: presa una terna di geni, esistono sei possibili permutazioni dell'ordinamento dei geni.

Nel caso in esame si ha che una di queste permutazioni è un'effettiva struttura FFL, trovata nella rete esaminata, mentre le rimanenti cinque sono istanze fittizie,

che non corrispondono cioè, a strutture FFL realmente presenti. Partendo da questa osservazione è possibile definire un primo semplice classificatore: data la sua maggiore numerosità, è possibile scegliere di classificare tutti i pattern nella classe negativa. In questo modo si migliorano le prestazioni di una scelta casuale, tra le due classi. Si ottiene, infatti, un classificatore di maggioranza, a cui corrisponde un valore di accuratezza fissa del $\frac{\lceil}{5}][6] = 83,333$ e un MCC pari a 0. Questo semplice classificatore, grazie alle prestazioni fisse, non dipendenti da quale dei dataset si usa per la validazione, sarà utilizzato come metro di paragone dei risultati ottenuti tramite il classificatore basato su SVM.
I risultati, ottenuti in questa fase, mostrano un comportamento sempre migliore, rispetto alle prestazioni del classificatore di maggioranza. Si nota anche, che il kernel lineare ha sempre prestazioni inferiori agli altri due.
Nell'inferenza delle FFL coerenti del primo tipo il kernel RBF ottimale è quello che mostra le prestazioni migliori, con un'accuratezza del 92,593% e un MCC del 89,037%. La situazione è molto simile nell'inferenza delle FFL incoerenti del terzo tipo, per cui risulta ancora migliore il kernel RBF, con un'accuratezza del 92,857% e un MCC del 88,604%. In entrambi questi casi i valori sono significativamente superiori a quelli attesi, indicando che il classificatore, basato su SVM, si comporta in maniera significativa.
Il caso delle FFL coerenti del quarto tipo mostra le prestazioni migliori con il kernel polinomiale, arrivando ad un'accuratezza del 91,667% e un MCC del 90,117%
I casi rimanenti mostrano un comportamento differente. I kernel polinomiali e RBF hanno prestazioni simili tra loro, anche se quello polinomiale risulta leggermente migliore; i valori di accuratezza e MCC, inoltre, sono dell'ordine di quelli del classificatore di maggioranza, quindi mostrano un comportamento meno significativo del classificatore. Nella classificazione delle FFL coerenti del terzo tipo si è ottenuta un'accuratezza del 85,556% e una MCC del 71,340%, mentre per le FFL incoerenti del primo tipo, un'accuratezza del 75% e una MCC del 78,028%.
I parametri testati riguardavano due tipologie di modifiche dell'algoritmo di classificazione:

1. Parametri interni del classificatore, definiscono aspetti del processo di classificazione, come ad esempio l'utilizzo di iperpiani con o senza bias;

2. Parametri interni del kernel, definiscono aspetti riguardanti la classe di funzioni utilizzata come kernel.

Nella determinazione dei parametri ottimali, si è iniziato ottimizzando la prima categoria, per passare in seguito alla seconda. Ricordando che il kernel polinomiale ha la seguente forma

$$k(x, y) = (s \cdot x \cdot y + c)^d$$

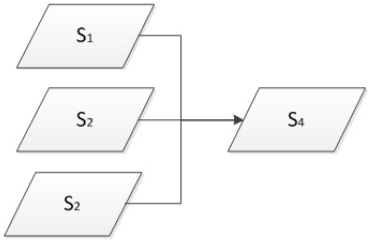


Figura 7.9: Struttura della fase di test del classificatore monoclasse.

nei tre casi presi in analisi, le configurazioni ottimali sono le seguenti:

1. Nella determinazione delle FFL coerenti del primo tipo si è ricavato $d = 7$;
2. Nella determinazione delle FFL coerenti del terzo tipo si è ricavato $d = 5$;
3. Nella determinazione delle FFL coerenti del quarto tipo si è ricavato $d = 9$;
4. Nella determinazione delle FFL incoerenti del primo tipo si è ricavato $d = 2$;
5. Nella determinazione delle FFL incoerenti del terzo tipo si è ricavato $d = 6$;

Non sono stati specificati i corrispettivi valori dei parametri s e c, in quanto la variazione di questi, rispetto a quelli utilizzati di default dal software, non ha comportato modifiche significative, alle prestazioni del classificatore.
Per quanto riguarda, invece, il kernel RBF, che ricordiamo è descritto dalla seguente formula

$$k(x, y) = \exp(-\gamma \|x - y\|^2)$$

sono stati determinati i seguenti valori ottimali di γ:

1. Nella determinazione delle FFL coerenti del primo tipo si è ricavato $\gamma = 4$;
2. Nella determinazione delle FFL coerenti del terzo tipo si è ricavato $\gamma = 7$;
3. Nella determinazione delle FFL coerenti del quarto tipo si è ricavato $\gamma = 9$;
4. Nella determinazione delle FFL incoerenti del primo tipo si è ricavato $\gamma = 9$;
5. Nella determinazione delle FFL incoerenti del terzo tipo si è ricavato $\gamma = 9$;

A questo punto si è passati alla fase di testing del classificatore, secondo lo schema presentato nella figura 7.9.
Di seguito sono riportati i risultati di questa fase, ottenuti utilizzando le configurazioni ottimali, precedentemente ricavate.

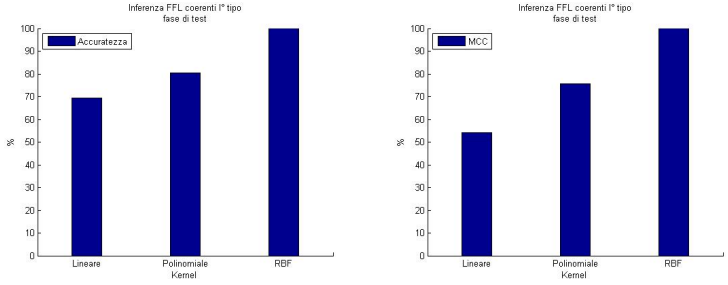


Figura 7.10: Risultati test dei FFL coerenti del primo tipo.

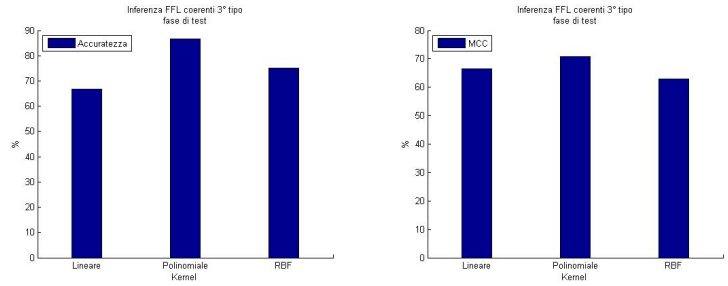


Figura 7.11: Risultati test dei FFL coerenti del terzo tipo.

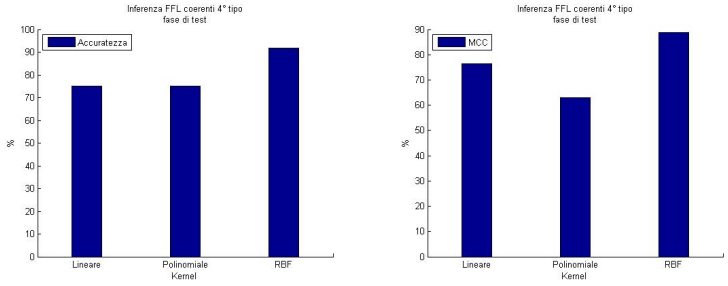


Figura 7.12: Risultati test dei FFL coerenti del quarto tipo.

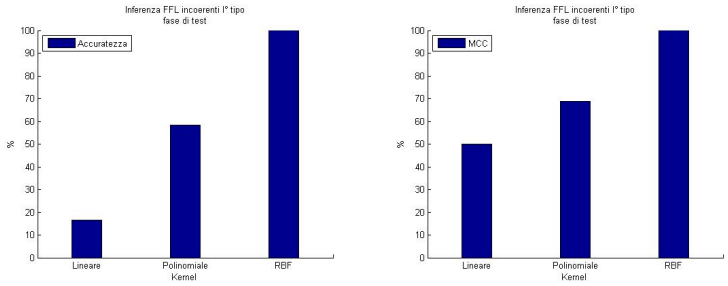


Figura 7.13: Risultati test dei FFL incoerenti del primo tipo.

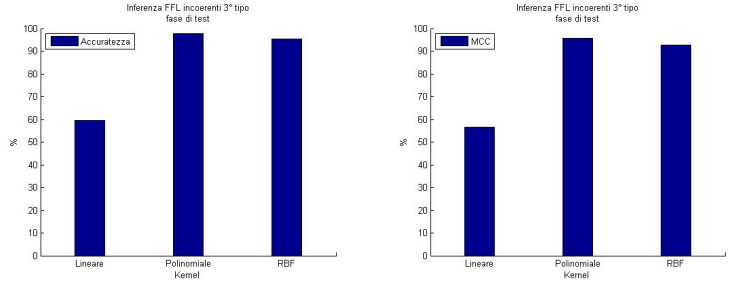


Figura 7.14: Risultati test dei FFL incoerenti del terzo tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	69,444	0,086	54,303
Polinomiale	80,555	0,512	75,619
RBF	100	1	100

Tabella 7.7: Risultati test dei FFL coerenti del primo tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	66,666	0,331	66,545
Polinomiale	86,666	0,418	70,916
RBF	75	0,258	62,909

Tabella 7.8: Risultati test dei FFL coerenti del terzo tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	75	0,529	76,457
Polinomiale	75	0,258	62,909
RBF	91,667	0,775	88,729

Tabella 7.9: Risultati test dei FFL coerenti del quarto tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	16,666	0	50
Polinomiale	58,333	0,378	68,898
RBF	100	1	100

Tabella 7.10: Risultati test dei FFL incoerenti del primo tipo.

Kernel	Accuratezza	MCC ass.	MCC perc.
Lineare	59,524	0,129	56,455
Polinomiale	97,619	0,913	95,643
RBF	95,238	0,856	92,817

Tabella 7.11: Risultati test dei FFL incoerenti del terzo tipo.

Le ottime prestazioni ottenute nella fase di training, per la determinazione delle FFL coerenti del primo tipo, sono state confermate pienamente nella successiva fase di test. Risulta, infatti, che la configurazione individuata prima come ottima, ha riconosciuto correttamente tutte le terne, realizzando un classificatore perfetto (accuratezza e MCC pari al 100%). Anche nel caso delle FFL incoerenti del terzo tipo, le buone prestazioni mostrate sono state confermate. Una particolarità in questa situazione è che, la configurazione che nella fase di test risulta migliore è quella polinomiale e non quella RBF, come accadeva per la fase di training. La differenza nelle prestazioni non è elevata, per cui si può dedurre

che i parametri del kernel RBF abbiano subito un overfitting sui dati di training, portando così ad un calo dei valori di accuratezza e di MCC nella fase di verifica. Complessivamente entrambi questi kernel mostrano comunque prestazioni poco discoste dal classificatore perfetto, quindi il comportamento complessivo è decisamente significativo.
Per quanto riguarda la determinazione delle FFL coerenti del terzo tipo, sono state confermate grossomodo le prestazioni della fase di training. La configurazione più efficiente è quella polinomiale, come accadeva per l'addestramento. Nello specifico si ha un leggero miglioramento del valore di accuratezza risultante, anche se la correlazione MCC rimane piuttosto bassa.
Le rimanenti due tipologie mostrano un comportamento discosto da quello ottenuto nelle relative fasi di training. Nella determinazione delle FFL coerenti del quarto tipo si ottiene una prestazione molto buona con il kernel RBF, mentre il kernel polinomiale, che era stato scelto come ottimale nell'addestramento, non ha confermato le prestazioni attese. Lo stesso sbilanciamento si ha con le FFL incoerenti del primo tipo, in cui, utilizzando il kernel RBF, si ottiene un classificatore perfetto. Il comportamento del classificatore SVM, con queste ultime due tipologie di FFL, è molto sensibile al variare dei parametri del kernel. La ragione è da trovarsi nell'esiguo numero d'istanze positive, presenti nel dataset.
Tutti i risultati, mostrati fino ad ora, sono stati ottenuti utilizzando le rappresentazioni numeriche dei profili genetici, come base di partenza per il processo di classificazione. Nel caso, invece, dell'utilizzo della rappresentazione simbolica, si è arrivati a dei risultati costanti, equivalenti al semplice classificatore di maggioranza. Nonostante nei test si sia fatto ricorso a tutte e tre le tipologie di kernel e si sia cercato di variare i rispettivi parametri, il comportamento del classificatore ha mantenuto un comportamento costante: tutte le terne sono state classificate come appartenenti alla classe negativa.
La motivazione di questi risultati, così poco significativi, può essere vista nell'eccessiva riduzione delle dimensioni dello spazio d'input. Il passaggio dai 21 campioni per geni, della rappresentazione numerica, ai 3 o 4 simboli, della rappresentazione simbolica, ha portato ad una forte perdita d'informazione, che il classificatore, basato sull'algoritmo SVM, ha tramutato in un comportamento molto elementare, nell'operazione di classificazione. Per questa ragione nella fase successiva di analisi del classificatore SVM multiclasse, la rappresentazione simbolica non è stata considerata.

7.4 Valutazione del Classificatore SVM Multiclasse

Una volta appurate le buone prestazioni ottenute, utilizzando un classificatore basato su SVM, per l'identificazione della singola tipologia di FFL, si è passati ad un problema più generale, vale a dire realizzare un classificatore multiclasse. I

risultati mostrati finora, dimostrano che, data una terna che rappresenta una FFL, presente nella rete, di cui non si conosce la tipologia di appartenenza, il classificatore, basato sulle SVM, è in grado di decidere l'effettiva tipologia d'appartenenza.
Con questo nuovo obiettivo è stata modificata la struttura dei dataset utilizzati: si è passati da dataset costituiti da pattern basati su di una singola tipologia di FFL, a dataset in cui sono presenti pattern che hanno origine da tipi diversi di FFL.
Nel caso del classificatore multiclasse si è fatto ricorso ad un'implementazione ad hoc delle SVM, $SVM^{multiclass}$ [Joa08b], realizzata sempre dal Dipartimento della Computer Science della Cornell University. La versione utilizzata è la 2.20 del 14.08.2008.
Coerentemente con quanto studiato finora, sono state mantenute invariate le tipologie di FFL analizzate. Si è scelto, però, di tralasciare le FFL coerenti del terzo tipo, poiché le prestazioni ottenute nella fase precedente, non sono risultate sufficientemente significative e che per questo il loro inserimento, in questa fase del lavoro, avrebbe potuto portare ad un peggioramento, nelle prestazioni generali del classificatore.
Una volta fissata la struttura del dataset utilizzato, sono state individuate due tecniche implementative:

1. Realizzare un classificatore per ogni tipologia ricercata;
2. Realizzare un unico classificatore multiclasse.

Per effettuare il processo di Machine Learning, è stato necessario definire una configurazione ottimale unica, per la determinazione delle diverse tipologie di FFL. Basandoci sui risultati ottenuti, si è scelto di valutare il comportamento medio delle configurazioni ottimali dei tre kernel: il kernel RBF è risultato quello che fornisce le prestazioni mediamente migliori. Per quanto riguarda il valore di γ si è trovato che con $\gamma = 9$ si ha un'accuratezza media del 93,953% e un MCC del 91,452%, nelle quattro tipologie di FFL prese in esame.
Una particolarità di questa configurazione è che, nonostante non si ottengano classificatori perfetti per le singole tipologie di FFL, come accadeva per le configurazioni ottimizzate per il singolo kernel, la prestazione media è comunque superiore. Una volta fissata la nuova configurazione dei parametri relativi al classificatore, la struttura del processo di Machine Learning è sempre quella riportata nella figura 7.9.
Come strumento aggiuntivo per la valutazione dei classificatori sono state utilizzate le curve ROC (Receiver Operating Characteristics). In un processo di reperimento dell'informazione è possibile visualizzare su di un piano cartesiano il rapporto tra le misure di precisione e di richiamo, descritte in precedenza. La curva che si ottiene, detta ROC, costituita sulle ascisse dalle misure del richiamo

e sulle ordinate dalle misure di precisione, sintetizza il comportamento del classificatore al variare del numero di pattern assegnati alla classe positiva, vale a dire quella delle terne di geni che rappresentano una struttura FFL effettivamente presente nella rete. Nella pratica, questa curva può essere utilizzata per ottenere il valore migliore di precisione del classificatore, in concomitanza ad un fissato valore di richiamo.
Prima di passare all'analisi delle due tipologie di classificatori multiclasse, ci si è posto un problema intermedio, vale a dire determinare la tipologia di appartenenza di una FFL, partendo da dataset costituiti da tutte e sole le istanze relative a terne di geni che costituiscono FFL effettivamente presenti nelle reti. Di per sé, questo problema è un ulteriore caso di classificatore multiclasse, semplificato rispetto al problema di partenza, dato che le dimensioni dei dataset in questione sono minori. Si ha, quindi, anche in questo caso, la possibilità di procedere in due modi: dato che le tipologie di FFL da identificare sono quattro, si possono realizzare quattro classificatori a singola classe o un classificatore con quattro classi. In entrambi i casi, per questo problema, si ottiene un comportamento equivalente, mostrato di seguito.

Tipologia FFL	**Accuratezza**	**MCC ass.**	**MCC perc.**	**Precisione**	**Richiamo**
1° tipo coerente	82,353	0,633	81,631	71,428	83,333
4° tipo coerente	100	1	100	100	100
1° tipo incoerente	100	1	100	100	
3° tipo incoerente	82,353	0,633	81,631	83,333	66,333

Tabella 7.12: Risultati del classificatore per l'identificazione della tipologia di FFL, tra le quattro prese in esame.

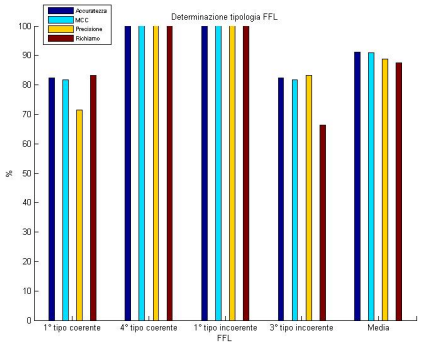


Figura 7.15: Risultati del classificatore per l'identificazione della tipologia di FFL, tra le quattro prese in esame.

I risultati mostrano che le classi del quarto tipo coerente e del primo tipo incoerente sono identificate perfettamente. Per quanto riguarda invece le FFL coerenti del primo tipo si ottiene un richiamo del 83,333% e una precisione del 71,428%, mentre per le FFL incoerenti del terzo tipo si arriva ad un richiamo del 66,333%

e una precisione del 71,428%. Dalla struttura dei risultati ottenuti è possibile formulare una strategia operativa, che si adatta al caso in esame. Questa prevede, in un primo tempo, di reperire le classi relative alle FFL coerenti del quarto tipo ed incoerenti del primo tipo, per poi eseguire un ulteriore processo di classificazione, solamente tra le rimanenti due classi. In quest'ultimo processo di classificazione si è ottenuta un'accuratezza e una MCC del 77%, un richiamo del 71,428% e una precisione del 83,333%.
A questo punto si è passati alla realizzazione del classificatore multiclasse vero e proprio. Come già accennato prima, i dataset utilizzati in questa fase sono costituiti da tutte le istanze delle quattro tipologie di FFL prese in esame e dalle relative permutazioni dell'ordinamento dei tre geni. Lo spazio di ricerca si è così esteso, rispetto al caso precedente, aumentando lo sbilanciamento dei dataset.

Di seguito sono riportati i risultati ottenuti.

Tipologia FFL	**Accuratezza**	**MCC ass.**	**MCC perc.**	**Precisione**	**Richiamo**
1° tipo coerente	92,158	0,627	81,339	60	100
4° tipo coerente	95,098	0,521	76,049	66,66	100
1° tipo incoerente	98,039	0,49	74,5	50	50
3° tipo incoerente	89,216	0,455	72,774	41,66	71,43

Tabella 7.13: Risultati del classificatore a classe singola per l'identificazione della quattro tipologie di FFL.

Tipologia FFL	**Accuratezza**	**MCC ass.**	**MCC perc.**	**Precisione**	**Richiamo**
1° tipo coerente	96,078	0,593	79,673	75	50
4° tipo coerente	100	1	100	100	100
1° tipo incoerente	99,019	0,704	85,180	80	66,67
3° tipo incoerente	97,059	0,753	85,759	100	50

Tabella 7.14: Risultati del classificatore multiclasse per l'identificazione delle tipologie di FFL.

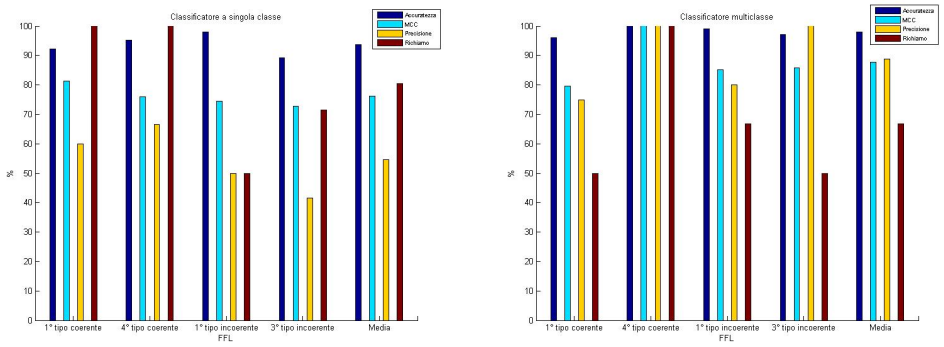


Figura 7.16: Risultati dei singolo classificatori, a sinistra e del classificatore multiclasse a destra, per l'identificazione delle tipologie di FFL.

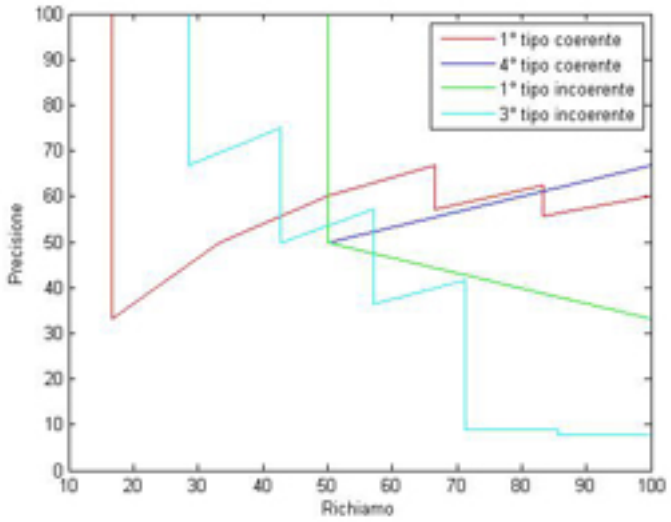


Figura 7.17: ROC del classificatore multiclasse.

Analizzando i risultati, nel caso dei singoli classificatori, si nota che i FFL coerenti del primo e del quarto tipo sono reperiti correttamente, insieme però a dei falsi positivi. Per le rimanenti due tipologie, le FFL correttamente reperite sono solo un sottoinsieme di tutte quelle effettivamente presenti. Con questa implementazione si è ottenuta un'accuratezza media del 93,627% e una MCC media del 76,166%.

Nel caso del classificatore multiclasse si è fatto ricorso a cinque classi, di cui quattro per le corrispondenti tipologie di FFL da identificare, più una aggiuntiva per le terne ottenute tramite l'operazione di permutazione, vale a dire le terne che non corrispondono a strutture FFL effettivamente presenti nelle reti.

Con quest'ultima configurazione si ottiene un'accuratezza media complessiva del 98,039% e una MCC media di 87,653%. Si ha quindi che l'implementazione più efficiente è quella ottenuta usando un singolo classificatore a cinque classi.

7.5 Conclusioni

L'obiettivo che ci si è prefissato è stata l'identificazione delle tipologie di FFL presenti nelle reti prese in esame. Come dataset di partenza si è fatto ricorso ad un insieme di dati, denominato dream3, costituito da 4 set per 5 reti, formate da 10 geni, campionati in 21 punti. Il primo passo è stato quello di considerare separatamente le tipologie di FFL, costruendo i dataset a partire dalle istanze della FFL realmente presenti. Sono state create due classi, una positiva ed una negativa, sfruttando la significatività dell'ordinamento dei geni.

A questo punto è stata confrontata quale rappresentazione dei profili genici, usata come base per il processo di classificazione, fornisce le prestazioni migliori. Si è visto che l'utilizzo della rappresentazione simbolica ha portato a risultati poco significativi, equivalenti a quelli ottenuti tramite un classificatore di maggioranza, che è un termine di paragone, nel nostro caso in esame.
Per questa ragione si è scelto di abbandonare questa rappresentazione e di continuare le successive fasi, facendo ricorso univocamente alla rappresentazione numerica. Con essa, si è realizzato un processo di Machine Learning, partendo da una procedura di 3-cross validation per l'ottimizzazione dei parametri dei kernel utilizzati, per poi testare queste configurazioni su altri dati. In questa fase di testing, mantenendo separate le tipologie di FFL, si sono ottenute prestazioni significative per quattro tipi su cinque. Nel caso meno rilevante si sono ottenuti risultati comunque migliori del classificatore di maggioranza, ma non di molto. Nei rimanenti casi l'accuratezza, invece, è significativamente superiore a quella attesa.
Grazie a questi buoni risultati si è deciso di estendere l'obiettivo, cominciando a considerare contemporaneamente più tipologie di FFL. Per la risoluzione di questo problema, sono state individuate due strategie risolutive: la prima consiste nel classificare una tipologia alla volta, mentre la seconda prevede un singolo classificatore per riconoscere contemporaneamente tutte le tipologie.
Queste due strategie sono state confrontate tra loro ed è risultato che, implementando un singolo classificatore, che riconosce contemporaneamente le tipologie di FFL, si ottengono prestazioni migliori, rispetto all'utilizzo di singoli classificatori. Nel caso ottimo si ha un'accuratezza media complessiva del 98,039% e una MCC media di 87,653%, che dimostrano un comportamento significativo del classificatore mediante SVM.

Appendices

Appendice A

Tecniche di Machine Learning

A.1 Introduzione

La determinazione delle reti di regolazione genica è un settore della ricerca scientifica che ha assunto un ruolo importante in quest'ultimo periodo. Molti algoritmi sono stati sviluppati a questo scopo, spesso prendendo concetti da altre discipline ed adattandoli a questo specifico caso.
Il problema generale può essere visto come un caso di *reverse engineering*, in quanto esso prevede, partendo da dei dati ottenuti sperimentalmente, di ricostruire il modello sottostante che li ha generati. Uno sviluppo di quest'area di ricerca è la capacità di prevedere il comportamento della cellula, in occasione di un determinato stimolo, partendo da un modello creato sperimentalmente. Un altro importante impiego è la determinazione di particolari configurazioni d'interazioni tra geni, che svolgono una funzione importante nella cellula, come accade, ad esempio, nel caso dei network motif.
In questa appendice saranno discussi, dopo una descrizione generale del processo di classificazione, i principali metodi di reverse engineering, utilizzati su dati genici, come ad esempio gli algoritmi di clustering, oppure quelli basati su di un approccio probabilistico, tipico delle reti bayesiane; sarà analizzato un algoritmo per l'elaborazione di serie temporali, vale a dire l'algoritmo DTW. Infine saranno trattate le tecniche basate sulla teoria dell'informazione, ed in particolare i due algoritmi che rappresentano lo stato attuale dell'arte, in questo settore: l'algoritmo REVEAL e l'algoritmo ARACNE.

A.2 Processo di Classificazione

Attualmente le tecniche di Machine Learning rappresentano lo stato dell'arte per la risoluzione di diverse problematiche, come accade per esempio nei seguenti

casi:

1. La definizione dell'algoritmo che risolve un problema risulta troppo difficile; in questo caso il processo di Machine Learning permette di costruire un modello, a partire da un set di dati, utilizzati nella fase di addestramento, per inferire regole su di altri dati;

2. L'applicazione richiede una fase di ottimizzazione dei propri parametri interni; in questo caso le tecniche di Machine Learning forniscono la possibilità di adattare un algoritmo all'ambito di lavoro.

Queste caratteristiche rimarcano quale sia l'importanza di questo settore, nell'ambito scientifico e il motivo per cui la ricerca si sia concentrata molto su di esso.
Gli obiettivi del processo di Machine Learning possono essere raggruppati nelle seguenti classi:

1. Classificazione, determinazione della classe di appartenenza di un set di dati;

2. Regressione, determinazione della funzione che genera un set di dati;

3. Data Mining, determinazione di nuova informazione significativa a partire da un ampio set di dati.

Un'ulteriore suddivisione degli algoritmi per l'apprendimento automatico, è la seguente:

1. Apprendimento supervisionato, nel quale l'intero procedimento si basa su di una fase di training, effettuata su di un sub set di dati, dei quali si conosce a priori la struttura significativa, come ad esempio, nel caso di un problema di classificazione, la classe di appartenenza dei dati;

2. Apprendimento non supervisionato, nel quale non è disponibile nessuna forma di conoscenza a priori sui dati da analizzare.

Nel caso di questo lavoro si è valutato tra l'altro il comportamento di un classificatore nel processo di apprendimento supervisionato di sottostrutture significative, presenti in alcune reti di regolazione genetica.
Nel campo di ricerca dell'apprendimento automatico, sono diversi gli argomenti di dibattito e di sviluppo. Ad esempio, l'utilizzo di dati non classificati, all'interno di metodi supervisionati, rappresenta una possibilità di ridurre i costi computazionali della fase di training dell'apprendimento. Un esempio è la costruzione di sistemi in gradi di fare un vero e proprio merging della conoscenza, derivante

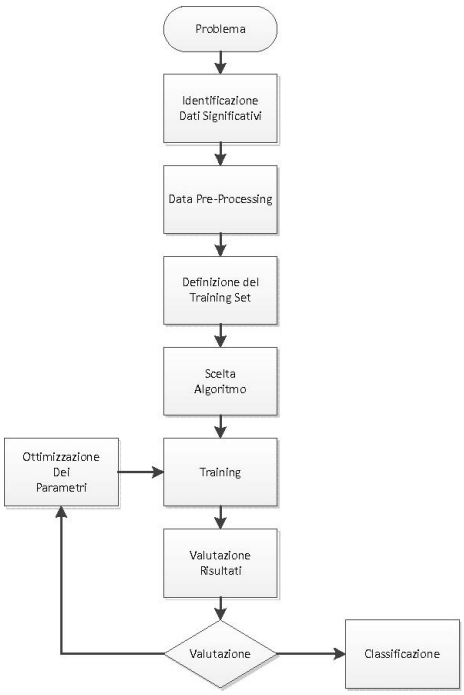


Figura A.1: Passaggi del processo di Machine Learning

da più processi di apprendimento. Grazie a questo, è possibile passare dalla determinazione di una singola funzione, relativa ad un insieme di dati, alla scoperta di una famiglia di funzioni, tra di loro collegate.
Un altro fondamentale problema della ricerca è la determinazione delle differenze tra gli algoritmi di Machine Learning. La questione si può equivalentemente porre come la scelta dell'algoritmo migliore, per una determinata funzione e in un preciso ambito.
Un'altra scelta, decisiva per l'effettiva efficienza dell'intero processo, è quella dell'insieme utilizzato come training e della strategia utilizzata per questa fase. Nel caso delle tecniche di data mining, un fattore importante, invece, è la definizione di una sorta di confine che preservi il diritto alla privacy dell'individuo.

A.3 Algoritmi di Clustering

Il clustering organizza i geni in gruppi (*cluster*) con pattern di espressione simili. Spesso i geni appartenenti allo stesso cluster sono detti coespressi. Le ragioni per cui si cercano geni coespressi sono le seguenti:

1. È stato dimostrato nella pratica che molti geni funzionalmente correlati, risultano coespressi; ad esempio, si ha che geni che codificano elementi di un complesso proteico hanno solitamente pattern di espressione simili;

2. Geni coespressi possono dare informazioni sui meccanismi regolatori; se un sistema regolativo controlla due o più geni, è probabile che questi risulteranno essere coespressi.

Differenti tecniche di clustering sono state sviluppate ed applicate all'analisi di dati provenienti da microarray. Il clustering raggruppa i geni che mostrano simile comportamento tra i differenti punti sperimentali. Bisogna tenere presente che questo metodo può essere utilizzato esclusivamente nel caso in cui i punti sperimentali siano almeno due, altrimenti il clustering diviene completamente inutile.

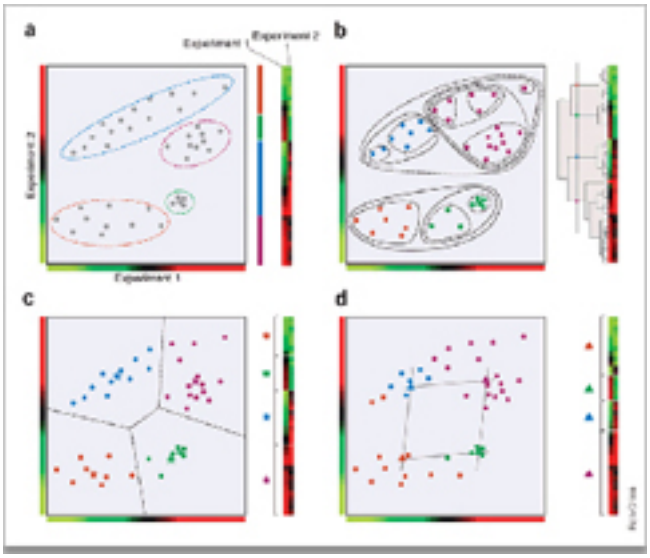


Figura A.2: Esempi di clustering su dati sperimentali.

Le tecniche di clustering sono basate sulla definizione di una *distance metrics*. Nel nostro caso, questa rappresenta una distanza misurata tra tutte le coppie delle espressioni dei geni. In base a questa metrica, vengono raggruppati nello stesso cluster tutti quei geni che mostrano distanza minima tra di loro, vale a dire sono raggruppati quei geni che risultano più vicini, dal punto di vista biologico. Anche per quel che riguarda l'operazione vera e propria di clustering, esistono numerosi algoritmi. I 2 tipi di clustering più utilizzati sono quello gerarchico e il K-Means. Nella classificazione e valutazione di questa classe di metodi, si utilizzano e si valutano una serie di proprietà, come ad esempio le seguenti:

1. Un clustering si dice gerarchico se la sua rappresentazione ha una struttura ad albero, in cui oltre ad essere messi in risalto i vari gruppi costituiti, si ha che, a partire da un nodo radice, che contiene l'intera gerarchia, si dirama l'albero intero, fino ad arrivare ai singoli nodi che rappresentano i rispettivi geni;

2. Un clustering si dice supervisionato, se nel processo di determinazione dei gruppi si utilizzano delle informazioni note a priori sui geni, come ad esempio accade nel clustering basato sulle reti neurali, in cui il processo vero e proprio di raggruppamento è preceduto da una fase di training dell'algoritmo su dati di cui sono note le caratteristiche principali; nel caso in cui sia nota nessuna informazione sui dati, si parla di clustering non supervisionato;

3. Un clustering è agglomerativo se si parte da un singolo nodo e si costituiscono i gruppi, in base alla misura della distanza; questi metodi procedono in maniera opposta rispetto a quelli gerarchici, che partono dall'insieme totale dei geni per poi suddividerlo mano a mano nei vari sottogruppi.

Per la natura di questi algoritmi è necessario precisare che è assolutamente indispensabile eseguire clustering solo dopo aver filtrato statisticamente i dati da analizzare; costruendo una serie di gruppi attorno ai geni dobbiamo evitare il più possibile di mantenere quei geni i cui valori risultano non affidabili, in modo da diminuire il rischio di costruire cluster errati.
Tutti gli algoritmi di clustering utilizzano una misura di somiglianza tra vettori, per comparare i pattern di espressione. La definizione di questa dipende dalla *Similarity distances* utilizzata nella determinazione della matrice delle distanze. Le più importanti utilizzate sono:

1. Pearson correlation coefficient;
2. Uncentered Pearson correlation coefficient;
3. Squared Pearson correlation coefficient;
4. Averaged dot product;
5. Cosine correlation coefficient;
6. Covariance;
7. Euclidian distance;
8. Manhattan distance;
9. Mutual information;
10. Spearman Rank-Order correlation;
11. Kendall's Tau.

Vediamo ora brevemente i passi di un algoritmo gerarchico:

1. Calcola le distanze tra tutte le coppie possibili di geni, costruendo così la matrice delle distanze. Ogni gene, all'inizio, rappresenta un cluster contenente solo se stesso;
2. Trova i 2 cluster r e s con la minima distanza tra loro;
3. Fondi i due cluster r ed s e rimpiazza r con il nuovo cluster;
4. Elimina il cluster s e ricalcola le distanze che sono state interessate dalla fusione;
5. Ripeti i passi 2, 3 e 4 finché il numero totale dei cluster non diviene 1, cioè finchè non sono stati presi in considerazione tutti i geni.

I *Support Trees* sono un'evoluzione del clustering gerarchico che calcola, oltre agli alberi ottenuti dall'algoritmo precedente, anche un *supporto statistico* per i nodi dell'albero. Il clustering gerarchico è un tipo di clustering agglomerativo e quindi, nel costruire i gruppi, l'algoritmo parte da un gene a caso e a poco a poco inizia a costruire i cluster. Il risultato finale, quindi, può essere dipendente dal gene scelto casualmente per primo. L'operazione di resampling non fa altro che ripetere la stessa operazione di clustering sullo stesso set di dati, ma partendo ogni volta da un gene diverso. In questo caso si può valutare quante volte un gruppo di geni viene clusterizzato assieme, in modo da ricavare una vera e propria misura dell'attendibilità del cluster preso in esame.
Il K-Means è un tipo di clustering utile quando l'agente ha già delle ipotesi sul numero effettivo dei cluster, che i geni dovrebbero costituire. K è infatti il numero di cluster che l'algoritmo costruisce partendo dai dati. Di seguito sono riportati in maniera descrittiva i passi di un algoritmo per questa classe di clustering.

1. Assegna tutti i geni di un esperimento ad uno dei k cluster;
2. Calcola la media o la mediana per ognuno dei cluster;
3. Calcola la distanza tra ogni oggetto e la media o la mediana di ogni cluster;
4. Sposta ogni oggetto nel cluster la cui media è più vicina a quell'oggetto;
5. Ricalcola media dei cluster interessati dalla riallocazione;
6. Ripeti le operazioni 3, 4 e 5 finchè non sono necessari più spostamenti o si è raggiunto il massimo di iterazioni consentite.

Un altro possibile approccio sono le *Self Organizing Map*, basate sulle reti neurali: l'algoritmo si allena su di una porzione dei dati e costruisce i gruppi in base al modello che si è creato nella fase di training. Una SOM è una griglia rettangolare o esagonale in cui ogni cluster è rappresentato da un elemento. I passi di un algoritmo per questa classe sono riportati di seguito.

1. Fase di training in cui si prende un gene a caso e si effettuano le misure di distanza di esso da tutti gli altri nodi della mappa. Si passa ai geni successivi modificando ogni volta i nodi della griglia;

2. Fase di clustering in cui lo spazio virtuale dell'esperimento viene ridotto e sovrapposto alla griglia ed ogni gene associato ad un vettore di essa.

Un forte limite di queste tecniche è che l'induzione per cui dei geni appartenenti allo stesso cluster sono tra loro collegati, nella realtà, può anche essere contraddetta. Esistono, infatti, dei casi in cui dei geni risultano fisicamente legati, tramite dei geni intermediari, mentre dal metodo di clustering, il fatto che risultino appartenenti allo stesso gruppo non descrive completamente questa situazione. Come si può notare questa è un'effettiva limitazione che, nella pratica, impedisce di ottenere un unico modello, a partire dalla raggruppazione dell'algoritmo di clustering.

A.4 Algoritmo DTW

L'algoritmo DTW (Dynamic Time Warping) cerca delle corrispondenze tra i punti delle serie, relative ai profili dei geni che devono essere analizzati. L'approccio generale si basa sulla considerazione che in queste sequenze il parametro da utilizzare è il tempo, per cui la variazione d'ampiezza di una serie può essere interpretata come una variazione dello sviluppo temporale del gene.
L'algoritmo misura la differenza tra ognuno dei punti di ogni coppia correlata e poi somma tutte queste distanze; il risultato é utilizzato come un indice, che testimonia la correlazione tra le due serie temporali. Se chiamiamo A e B le due serie, che supponiamo avere lunghezza comune n, l'algoritmo crea una matrice $n \cdot n$, dove il valore di ogni punto $P[i, j]$ é la distanza tra il punto i della serie A e il punto j della serie B.
Dopo di ció, viene eseguito un algoritmo di ricerca del percorso minimo tra due punti, vale a dire tra il punto $[0, 0]$ e il punto $[n, n]$. La somma dei valori dei punti che si trovano sul percorso é il risultato della DTW. In pratica si ha che lo scopo di questo algoritmo è quello di ottenere il miglior mapping possibile fra le due successioni, in modo da minimizzare la loro distanza reciproca.
La denominazione *Time Warping* relativa a questa classe di algoritmi, deriva dal fatto che le serie di dati, utilizzate come input, sono serie temporali e che il mapping porta a delle operazioni di compressione o di espansione temporale. Questa caratteristica operazione di warping permette di ottenere una misura di somiglianza significativa, se riferita, ad esempio, ad una normale misura di distanza tra le due serie.
La proprietà di non linearità dell'allineamento della DTW permette di determinare più efficientemente la rassomiglianza, ad esempio in caso di eventuali ri-

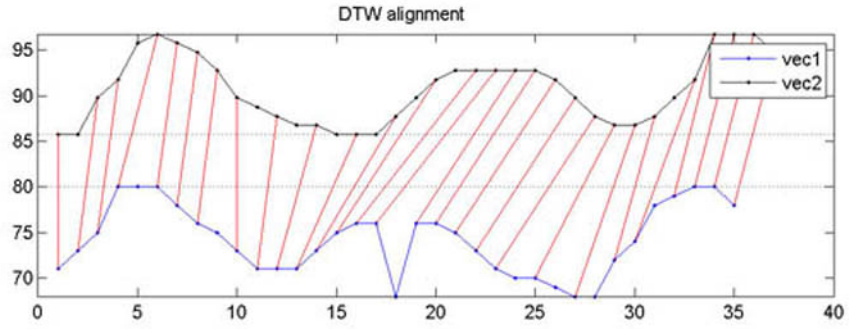


Figura A.3: Esempio di applicazione dell'algoritmo DTW ad una coppia di profili genici.

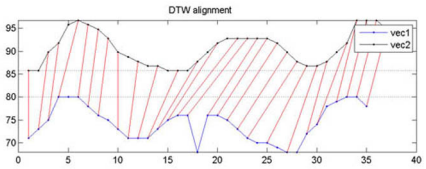
tardi o sfasamenti, in modo da poter determinare anche una relazione di causalità temporale tra i rispettivi geni.

Nella ricerca del migliore allineamento sono molte le proprietà che possono essere considerate, in modo da diminuire il campo di ricerca di tutti gli eventuali cammini possibili, che altrimenti sarebbero in numero esponenziale, rispetto alla lunghezza delle serie. Tra queste proprietà ricercate si possono considerare la monotonicità, per cui si ha che l'allinemento non andrà mai indietro nel tempo, in modo ad esempio, di evitare l'allineamento di una porzione di un segnale con più parti dell'altro segnale.

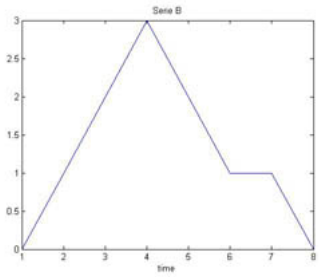
Un'altra importante proprietà ricercata è quella della continuità, per cui risulta che l'allineamento deve sempre prendere in considerazione tutte le porzioni significative di un segnale. Un'estensione, per certi versi, di questa proprietà è l'applicazione delle condizioni ai limiti, per cui risulta che tutto il segnale sia preso in considerazione e non solo una sua porzione.

Per quanto riguarda la determinazione pratica delle porzioni significative di un segnale, occorre far notare che la maggiore rassomiglianza si ha con un matching diretto senza operazioni di warping; in generale si ha, quindi, che l'andamento previsto dell'allineamento tenderà ad una serie di finestre di matching diretti, separate da delle operazioni di espansione o contrazione temporale.

La pecca di quest'algoritmo é che il costo computazionale é quadratico, $O\left(n^2\right)$, quindi per serie lunghe diventa molto costoso eseguire una DTW. Per ovviare a questo problema esistono opportune varianti dell'algoritmo DTW, come ad esempio, quella proposta da Stan Salvador e Philip Chan in "FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space", che prevede di eseguire una sequenza di DTW a risoluzione sempre maggiore, utilizzando ad ogni iterazione solo la parte di path evidenziata dall'iterazione precedente.



(a)



(b)

Figura A.4: Esempi di matrici di adiacenza dell'algoritmo DTW.

A.5 Approcci Basati sulla Teoria dell'Informazione

Prima di iniziare ad approfondire questa classe di algoritmi, è necessario introdurre alcune importanti definizioni.
Si definisce entropia di Shannon, la probabilità di osservare un certo evento in una data sequenza. Dal punto di vista del calcolo si ha che, poste p_i le probabilità che una variabile aleatoria X possa assumere i valori $1, 2, \ldots, n$, l'entropia può essere calcolata tramite la seguente formula

$$H = -\sum p_i \log p_i$$

Nel caso più semplice da analizzare, vale a dire con X variabile binaria, la funzione H ha un massimo in corrispondenza del caso in cui entrambi gli stati siano equiprobabili; nel caso, invece, in cui uno dei due stati divenga più probabile, la funzione H tenderà a decrescere.
Si parla di entropia individuale quando si ha a che fare con un'unica variabile aleatoria. Quando invece sono presenti più variabili, si parla di entropia combinata. Questa si calcola secondo la seguente definizione

$$H(x, y) = -\sum p_{i,j} \log (p_{i,j})$$

dove i $p_{i,j}$ sono le probabilità congiunte degli eventi i e j.
Come accade per la teoria delle probabilità, anche per l'entropia è possibile definire una forma condizionata.

$$H(X, Y) = H(X \mid Y) + H(Y) = H(Y \mid X) + H(X)$$

A questo punto è possibile introdurre il concetto di mutua informazione, che può essere interpretata come l'informazione condivisa tra due variabili aleatorie e che si calcola con la seguente formula

$$M\left(X,Y\right) = H\left(Y\right) - H\left(X \mid Y\right) = H\left(X\right) - H(Y|X)$$

oppure

$$M\left(X,Y\right) = H\left(X\right) + H\left(Y\right) - H\left(X,Y\right)$$

Questa espressione, nella pratica, è utilizzata comunemente per confrontare gli andamenti dei profili di geni, in modo da terminare un indice di correlazione tra di essi. In particolare, si utilizza la seguente proprietà:
la mutua informazione tra due geni i e j è uguale a zero quando la distribuzione congiunta dei valori delle espressioni dei due geni, non fornisce maggiore informazione delle singole distribuzioni, considerate separatamente; risulta invece maggiore di zero quando i loro profili sono associati in qualche modo.
Come si può facilmente notare da questa proprietà, la mutua informazione fornisce un indice di dipendenza tra due geni, per cui più questa risulta alta, più si è propensi ad ipotizzare una correlazione tra i geni esaminati.
Una sorta di svantaggio di questo metodo è che il grafo di dipendenze che viene determinato è non orientato, per cui non si è in grado di determinare con esattezza il verso della dipendenza, ma semplicemente vengono identificate le coppie di geni legate tra loro, senza specificare l'effettiva relazione di causa/effetto.

A.6 L'algoritmo REVEAL

REVEAL (REVerse Engineering ALgorithm) è un algoritmo per il reverse engineering di una rete di regolazione genetica. Questo si basa sulle reti booleane, per modellare le reti genetiche: i geni corrispondono ai nodi, mentre un arco orientato indica una relazione funzionale di causa ed effetto tra due geni. Nel dettaglio si ha che ciascun gene è visto come una variabile booleana in cui gli stati possibili sono acceso e spento, corrispondenti agli stati di attivazione e di dei profili dei geni.
L'idea alla base di questo algoritmo è quella di calcolare la mutua informazione, in modo da ricavare le relazioni tra geni. REVEAL, in particolare cerca la seguente partizione dei geni, per cui vale la seguente proprietà

$$\frac{M\left(X_0,S\right)}{H(X_0)} = 1$$

con S, un opportuno sottoinsieme dei geni. Quando vale

$$H\left(X\right) = M\left(X,Y\right) \Leftrightarrow H\left(Y\right) = M(X,Y)$$

si ha, per la definizione di mutua informazione, che risulta $H(X \mid Y) = 0$, vale a dire che Y determina completamente X. L'algoritmo REVEAL sfrutta questa proprietà per calcolare tutti i nodi che puntano ad un determinato altro nodo. Si cerca di ricavare se una variabile booleana X_0 dipenda dalle variabili $(X_1, \ldots, X_k, \ k \geq 2)$, secondo un'opportuna funzione booleana. REVEAL risolve il problema calcolando la mutua entropia

$$M(X_0, [X_1, \ldots, X_k]) = H(X_0) + H(X_1, \ldots, X_k) - H(X_0, X_1, \ldots, X_k)$$

e verificando se risulta

$$M(X_0, [X_1, \ldots, X_k]) = H(X_0)$$

o in altre parole se vale

$$H(X_1, \ldots, X_k) = H(X_0, X_1, \ldots, X_k)$$

L'algoritmo REVEAL Itera questo procedimento in base al valore del parametro k, che rappresenta la numerosità del sottoinsieme di geni, che determinano lo stesso. Nel caso in cui $k = 1$, vale a dire quando si prende in considerazione un unico gene come causa, ha poco interesse la scelta della funzione booleana per la valutazione del rapporto di causalità, anche perché è possibile utilizzare un unico operatore, la negazione. Quando invece si comincia ad aumentare la dimensione di questo insieme, si ha la necessità di determinare delle opportune funzioni booleane. Limitando l'esame al caso $k = 2$, si è notato che, sperimentalmente, chiamando X e Y le due variabili, le loro combinazioni, che risultano statisticamente più significative sono $X \wedge \overline{Y}$ e $\overline{X} \wedge Y$ (X and not Y e not X and Y).

A.7 L'algoritmo Aracne

L'algoritmo ARACNE (Algorithm for the Reconstruction of Accurate Cellular NEtwork) appartiene alla classe dei metodi che sfruttano la teoria dell'informazione, per l'inferenza delle reti di regolazione genetica. Come accade per REVEAL, è basato su di una misura della correlazione esistente tra coppie di geni. Allo stesso tempo si ha un'importante differenza, vale a dire che il grafo ricavato con l'algoritmo ARACNE è indiretto, in quanto questo metodo non determina il verso della dipendenza ma solo l'esistenza.
ARACNE si struttura in due passi. Per prima cosa è calcolata la mutua informazione tra ogni coppia di geni, in modo da determinare quali sono quelli che, dal punto di vista probabilistico, risultano essere maggiormente collegati. Per questo calcolo ci si trova in una situazione differente rispetto all'algoritmo REVEAL, in quanto le variabili aleatorie sono supposte continue, al contrario del caso precedente, in cui erano considerate discrete e, più nello specifico, binarie.

L'assunzione di variabili aleatorie continue porta ad una differente formula per il calcolo della mutua informazione, che è espressa dalla seguente espressione

$$M\left(X,Y\right)=\iint f\left(X,Y\right)\log\frac{f(X,Y)}{f\left(X\right)f(Y)}\partial X\partial Y$$

in cui $f\left(X\right)$ e $f(Y)$ sono le stime delle singole distribuzioni dei geni X e Y, mentre $f\left(X,Y\right)$ è la stima della distribuzione congiunta.
A questo punto c'è una fase di scrematura delle coppie che hanno una mutua informazione inferiore ad una determinata soglia, fissata a priori oppure determinata con tecniche d'apprendimento automatico o euristiche. Come accadeva con il calcolo della correlazione, anche in questo caso è possibile che geni non direttamente collegati, ma che sono in relazione, grazie a degli intermediari, risultino legati a questo punto dell'algoritmo.
Per questa ragione nel secondo passo dell'algoritmo si procede all'eliminazione delle relazioni giudicate non significative. Si ricorre ad una proprietà della mutua informazione, la DPI (Data Processing Inequality). La DPI dimostra che se due geni, g_1 e g_3 interagiscono solo tramite un gene intermediario g_2 allora vale la seguente disequazione

$$M(g_1,g_3)\leq min\left\{M\left(g_1,g_2\right),M(g_2,g_3)\right\}$$

Questa condizione è necessaria ma non sufficiente, perché risulta verificata anche se i geni g_1 e g_3 sono effettivamente collegati.
L'algoritmo ARACNE parte da un grafo, dove ogni lato rappresenta una coppia di geni la cui mutua informazione supera la soglia precedentemente fissata. In seguito viene esaminata ogni possibile tripletta, la cui mutua informazione supera la soglia e vengono eliminati i lati di questa tripletta che hanno un valore inferiore di un limite prefissato, questo senza tenere conto di precedenti eliminazioni in altre triplette.
Grazie a ciò la rete che viene ricostruita è indipendente dall'ordine con cui vengono esaminate le triplette. Sperimentalmente si nota che questa tecnica permette in maniera efficiente di rilevare strutture ad albero, in quanto le relazioni trovate hanno un'alta probabilità di rappresentare regolazioni dirette tra geni, senza però specificarne il verso nella relazione causa effetto, come invece accadeva nel caso dell'algoritmo REVEAL.

Elenco delle figure

Elenco delle tabelle

Bibliografia

[Alo07] Uri Alon. Network motifs: theory and experimental approaches. *Nature Publishing Group*, 2007. [cited at p. 12, 13, 63]

[BDC05] Claudio Cobelli e al Barbara Di Camillo, Gianna Toffolo. A quantization method based on threshold optimization for microarray short time series. *BMC Bioinformatics*, 2005. [cited at p. 37]

[Fal09] Marco Falda. Symbolic representations for reasoning about temporal gene profiles. Technical report, Dept. of Information Engineering, 2009. [cited at p. 38, 55]

[Gar03] Martina Garlet. Analisi dei livelli di espressione genica per lo studio delle lucemie. Master's thesis, Università degli studi di Padova, Facoltà di Scienze Statistiche, Corso di Laurea in Scienze Statistiche ed Economiche, 2003. [cited at p. 37]

[Gre09] Daiana Gressani. Confronto tra statistiche per l'ordinamento e la classificazione dei geni differenzialmente espressi. Master's thesis, Università degli studi di Padova, Facoltà di Scienze Statistiche, Corso di Laurea in Statistica e Informatica, 2009. [cited at p. 37]

[III04] Hal Daumé III. *SVMsequel Tutorial Manual*, 2004. [cited at p. 68]

[Joa08a] Thorsten Joachims. *SVMlight Support Vector Machine*, 2008. [cited at p. 68]

[Joa08b] Thorsten Joachims. *SVMmulticlass Support Vector Machine*, 2008. [cited at p. 80]

[LIKG93] Snezhana N. Neshkova Ludmila I. Kuncheva, Roumen Z. Zlatev and Hans Gamper. A combination scheme of artificial intelligence and fuzzy pattern recognition in medical diagnosis. Technical report, Central Laboratory of Bioinstrumentation and Automation, Sofia Bulgaria, 1993. [cited at p. 37]

[LS05] C. Larizza e al. L. Sacchi, R. Bellazzi. Taclustering: Cluster analysis of gene expression profiles through temporal abstractions. *Int. J. Med. Inform*, 2005. [cited at p. 38, 55]

[Mar08] Lorenzo Maragoni. Combinazione di modelli grafici per l'analisi di network biologici. Master's thesis, Università degli studi di Padova, Facoltà di Scienze Statistiche, Corso di Laurea Specialistica in Scienze Statistiche, Demografiche e Sociali, 2008. [cited at p. 37]

[Ris08] Davide Risso. Analisi dei dati di espressione genica: Studio comparativo per la valutazione dell'impatto della normalizzazione sull'inferenza statistica. Master's thesis, Università degli studi di Padova, Facoltà di Scienze Statistiche, Corso di Laurea Specialistica in Statistica ed Informatica Gestione delle Imprese, 2008. [cited at p. 37]