Sacri Appunti Intelligenza Artificiale 2

Riccardo Pietrucci

20/6/2006

Indice

1 Intelligent Web & Personalized Search

1.1 Information Overload

La montagna di pubblicazioni scientifiche è in costante aumento. Ma è tangibile la preoccupazione di impantanarsi nelle ricerche a causa del crescente numero di possibili specializzazioni. La difficoltà non è nella eccessiva mole di pubblicazioni ma piuttosto nella nostra abilità di gestirle proficuamente.

Possibile soluzione: personalizzare la Human-Computer interaction filtrando e proponendo solo le risorse di cui l'utente ha bisogno in un determinato momento.

Lo stato dell'arte dell'information filtering è il seguente:

Processo	Bisogno Informativo	Tipo di Sorgente
Information Filtering – IF	Stabile Specifico	Dinamica Non Strutturata
Information Retrieval - IR	Dinamico Specifico	Stabile Non Strutturata
Database Access	Dinamico Specifico	Stabile Strutturata
Exploration	Vario	Varia

IF vede il bisogno informativo statico dell'utente o lentamente variabile. E' possibile pensare di rappresentarlo internamente al sistema.

Le tecniche di **User modeling** consentono di identificare le caratteristiche importanti di un utente per potergli fornire informazioni più rilevanti.

1.2 Motori di ricerca

I **motori di ricerca** sono sistemi che utilizzano ricerche nel campo del IR. Il crawler si occupa di visitare risorse (html, pdf, etc.) in rete per il successivo processamento. L'obiettivo è salvare una copia dell'intero Web, e modificare la rappresentazione delle risorse in modo da eseguire le query molto velocemente ($<$1sec).

1.2.1 Inverted Index

Gli **indici** ci permettono di trovare le keyword di una query senza analizzare ognivolta tutte le pagine.

Quello che ci serve è trovare l'insieme di documenti che contengono una o più parole. Si usa l'**Inverted Index**, ovvero una lista ordinata di termini, con associate le informazioni relative ai documenti in cui compaiono.

1.2.2 Matching Vector space model

Per individuare i documenti che soddisfano una query si può utilizzare il *Boolean query processing*.

Ma se ci sono molti documenti che soddisfano la query è difficile per l'utente individuare il migliore. Allora si opera un *ranking*, ovvero si assegna un valore di similarità ad ogni documento e si propone una lista ordinata. Un possibile metodo per ricavare tale similarità è utilizzare il **Vector space model**:

- la query e i documenti vengono rappresentati da vettori in uno spazio dove le dimensioni fanno riferimento alle keyword
- la similarità è funzione della similarità tra vettori (e.g. prodotto scalare o differenza)

Come determinare i valori degli elementi dei vettori (*weighting*)? Possiamo pensare di rappresentare la query e i documenti come un insieme di parole (rappresentazione *bag of words*) e associare il numero di occorrenze tf (*term frequency*) ad ogni elemento del vettore.

Se prendiamo solo la frequenza rischiamo di pesare troppo i termini comuni che rappresentano poco un doc (avverbi,articoli,etc.). Allora possiamo determinare il peso di ogni termine j in un documento x in questo modo:

$$d_{x,f} = tf_{x,j} \cdot \log\left(\frac{N}{df_j}\right)$$

dove N è la dimensione della collezione predefinita e df_j (*document frequency*) è il numero di occorrenze del termine j all'interno della collezione. Tale tecnica si chiama **TFxIDF** (term frequency x inverse document frequency).

Infine si ricava la similarità Solitamente si utilizza la **cosine rule**. Ad esempio, dati i termini T_j, il vettore tridimensionale di esempio $v_1 = 2T_1 + 3T_2 + 5T_3$ e un secondo vettore analogo v_2, si calcola la similitudine attraverso il calcolo del coseno dell'angolo tra due vettori. dove il prodotto scalare si normalizza rispetto alla lunghezza dei vettori, in modo da rendere la similarità indipendente dalla lunghezza (misurata con il numero termini distinti) dei documenti.

1.2.3 Google

L'architettura di Google è così organizzata:

1. l'**URLServer** invia le urls da visitare ai crawler
2. lo **StoreServer** le compatta (zlib RFC1950) e le memorizza nel **Repositoy**
3. l'**Indexer** per ogni pagina estrae una lista di hits (parole+pos+font+etc) che viene salvate nei **Barrels** che contengono un *forward index* (per ogni doc la lista degli hits), e i *link*; le parole vengono inserite nel vocabolario **Lexicon**.
4. il **Sorter** crea l'inverted index
5. il **Searcher** esegue la query

1.3 Personalized Search

Utilizzare un modello degli interessi dell'utente per rendere una ricerca più precisa e aumentare il numero di documenti di interesse. Nel 99% dei casi esiste un modulo di User Modeling che contiene una rappresentazione dei concetti di interesse per l'utente, con un certo input per costruirlo e tenerlo aggiornato. Come vantaggi aiuta a risolvere problemi di polisemia e sinonimia, e può fornire adattabilità.

1.3.1 Google Alerts

L'utente suggerisce esplicitamente i termini di cui è interessato Il motore lancia periodicamente la query su News e/o Web e i risultati vengono inviati via email. Utile per bisogni informativi molto stabili. Nessuna adattività.

1.3.2 Approcci

Esistono due possibili tipi di dati:

User Data informazioni su caratteristiche personali dell'utente

Usage data informazioni sulle interazioni dell'utente

Per costruire e tenere aggiornato lo User Model c'è bisogno di un feedback esterno.

Explicit (Relevance) Feedback l'utente suggerisce documenti/parole di interesse

Implicit Feedback il sistema monitorizza il comportamento dell'utente

Una tecnica correlata al Explicit Feedback è la Query expansion: un sottoinsieme di termini estratti dai documenti vengono aggiunti alla query, incrementando (di solito) la precisione dei risultati.

Gli user profile possono essere sfruttati in due modi:

Part of retrieval process il ranking è un processo unificato. in cui gli user profile sono impiegati per il punteggio dei contenuti Web (più veloce, ma user model semplificati)

Re-ranking : gli user profile vengono impiegati in un secondo step, dopo che il punteggio è stato calcolato da un metodo non personalizzabile (migliori risultati, ma occorre rianalizzare tutti i documenti)

I possibili approcci disponibili:

Content-based invia raccondazioni all'utente (come gli IR tradizionali)

- **Current Context** si analizza il contesto dell'utente, come applicazioni aperte, documenti visualizzati, testo immesso, etc.
- **Search History** La browsing/query history può disambiguare il termine "Visa": se l'utente ultimamente ha cercato voli per un paese straniero, Visa riguarderà procedura purocratiche.
- **Rich User Models** UM con rappresentazioni più complesse dei needs (ad es. reti neurali)
- **Hypertextual Data** Versioni personalizzate di algoritmi che assegnano un rank alle pagine Web in base alla struttura dei links, e.g., PageRank, HITS.
- **Content Presentation** organizzare la lista dei risultati in cluster (grafici) contenenti documenti affini a un certo topic (Vivisimo)

Collaborative-based calcola le similarità (come Amazon)

- **Collaborative Approach** si suggeriscono documenti che altri con gli stessi needs (query) hanno selezionato in passato

2 Text Categorization

La categorizzazione prende in input: una descrizione di una istanza, $x \in X$, dove X è l'istanza linguaggio o spazio dell'istanza, un numero fissato di categorie $C = \{c_1, c_2, \ldots c_n\}$. In output: la categoria di x: $c(x) \in C$, dove $c(x)$ è una funzione di categorizzazione che ha come dominio X e come codominio C.

L'esempio di apprendimento è espresso con la coppia $< x, c(x) >$. Dato un insieme di esempi di apprendimento D, trovare una ipotizzata funzione di categorizzazione $h(x)$ tale che:

$$\forall < x, c(x) > \in D : h(x) = c(x)$$

Il Text Categorization assegna documenti ad un insieme fissato di categorie. Una categoria può essere rappresentata attraverso (ad esempio) con il TF/IDF.

2.1 Algoritmo di apprendimento Rocchio

Usiamo lo standard di indicizzazione TF/IDF per rappresentare in forma vettoriale i documenti di testo (normalizzati secondo la frequenza massima di un termine).

Per ogni categoria, viene elaborato un vettore "Prototipo" dalla somma dei vettori di training nella categoria.

Assegnamo il documento di test alla categoria col vettore "prototipo" più vicino mediante la regola di similarità del coseno.

2.2 Algoritmo di apprendimento Nearest-Neighbor

L'apprendimento si riduce al modo di immagazzinare le rappresentazioni degli esempi di training in D. Il test dell'istanza x:

- elabora la similarità tra x e tutti gli esempi in D.
- assegna ad x la categoria del più simile in D.

Quindi, a differenza del Rocchio, non si calcolano esplicitamente i prototipi delle categorie. I prototipi del Rocchio possono avere problemi con categorie disgiunte, mentre Nearest Neighbor tende ad avere in tal caso un comportamento migliore.

2.3 Algoritmo di apprendimento Bayesiano

Consiste nell'apprendere e classificare mediante approcci probabilistici. Il teorema di Bayes gioca un ruolo critico nell'apprendimento e classificazione.

Sapendo che la probabilità di A condizionata a B è pari a:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

il *teorema di Bayes* esprime la probabilità condizionata rispetto allo spazio degli eventi $A_1, A_2, \ldots, A_n$:

$$P(A_1|B) = \frac{P(B|A_1)P(A_1)}{P(B)} = \frac{P(B|A_1)P(A_1)}{\sum_{i=1}^{n} P(B|A_i)P(A_i)}$$

Dati l'insieme delle categorie $\{c_1, c_2, \ldots c_n\}$ ed una descrizione di un'istanza, determina il grado di appartenenza di E per ogni c_i. $P(E)$ può essere determinata solo se le categorie sono complete e disgiunte.

2.4 Reti Neurali

Una rete neuronale consiste in un pool di semplici processi elementari che comunicano fra loro spedendosi segnali attraverso numerose connessioni pesate. Una rete a strato singolo consiste in uno o più neuroni di output, ognuno dei quali è connesso con un fattore peso w_{ij} a tutti gli input x_i. In questa semplice rete (il neurone) può essere usato per separare gli input in due classi. I pesi della rete neurale sono modificati durante la fase di learning

Si utilizza un percettrone per ogni categoria Learning sui documenti di training della sua categoria. Durante la fase di test, il percettrone fornisce un valore VERO/FALSO sull'appartenenza del vettore rappresentativo il documento alla categoria.

2.5 Valutare la categorizzazione

Esistono due parametri accettati dalla comunità IR:

Precision l'abilità nel restituire i documenti che sono più rilevanti

Recall l'abilità nel restituire tutti i documenti rilevanti nell'intero dominio

2.6 Be MoRe

Be MoRe (Best Model Retrieval) è una nuova metodologia di classificazione di testi.

Il preprocessing prevede diverse fasi:

Suddivisione in Token L'NLP Natural Language Processing consiste nell'analizzare una frase in linguaggio naturale, eseguire il parsing ed applicare lo stemming, fino ad estrarne i nomi e i verbi. Il sistema, mediante il dizionario WordNet, assegna ad ogni significato un codice numerico univoco. Dopo aver eseguito gli algoritmi di word sense disambiguation, ad ogni nome della frase viene sostituito il codice del significato che è risultato essere migliore.

Compressione prevede due fasi:

1. si applica la TFxIDF per minimizzare il rumore
2. si applica la Edit Distance per calcolare la somiglianza morfologica (su Google "forse cercavi:")

La dimensione dello spazio dei termini può costituire un problema perché gli algoritmi di learning non scalano facilmente su grandi valori della dimensione. Se la dimensione è alta spesso si verificano fenomeni di overfitting. Abbiamo due scelte:

- Riduzione locale (un insieme di termini diverso per ciascuna categoria)
- Riduzione globale (il set di termini è valido per qualunque categoria)

Alla fine dovremo trovare l'iperpiano separatore ottimo dell'insieme di training. Come formalizzare? In due dimensioni, l'equazione della linea è data da: $w_1x + w_2y = b$ Se l'iperpiano separatore non esiste, ovvero se i dati non sono linearmente separabili per la presenza di rumore, si possono usare le Slack variables, che consentono la classificazione non corretta di alcuni punti, tenendo conto del rumore nei dati

L'OnLine Hyperplane consente la ricerca incrementale della soluzione ottima. Gli elementi positivi e negativi sono rappresentati con pesi diversi. È una Loss function a basso costo computazionale La convergenza è garantita dall'estensione del teorema di Novikoff.

Ereditando alcune caratteristiche matematiche delle SVM, il sistema si avvale di un modulo di kernel per risolvere il problema di separabilità non lineare.Scelta del kernel migliore attraverso model selection. L'uso della funzione kernel consente di calcolare l'iperpiano di separazione senza bisogno di effettuare esplicitamente il mapping in F

3 Focused Crawling

3.1 Introduzione

La dinamicità e le dimensioni del Web non ci permettono di costruire grandi basi di informazioni aggiornate su qualsiasi argomento (motori di ricerca) capaci di soddisfare efficacemente e velocemente qualsiasi query. I normali crawler terminano periodicamente l'esplorazione e ricominciano da capo per tenere aggiornate le copie delle pagine Web.

Obiettivo: partendo da una serie di pagine di partenza, navigando attraverso i link, scegliere di volta in volta i percorsi giudicati migliori, riducendo le risorse necessarie (cpu e network) per analizzare tutte le pagine, evitando di seguire i percorsi che lo portano a pagine non affini alla nostra query. Vantaggi: più coverage sulle risorse di interesse, più freshness nei risultati, e un matching più sofisticato.

Il *Focused (o Intelligent) Crawling* può essere usato per costruire indici inversi su certi topic in alternativa a

meta-searching interroga motori di ricerca esistenti

query-time crawling avviare crawling sul Web ad ogni query

query-modification non si costruisce un indice ma si modifica la query e la si inoltra a motori di ricerca di esistenza

Nella pratica si deve stabilire l'ordine delle prossime pagine da visitare, in modo da indirizzare l'esplorazione sempre verso i percorsi più interessanti Informazioni da sfruttare:

1. contenuto pagine visitate
2. ancore testuali dei link nelle pagine
3. struttura dei link tra pagine

3.2 Analisi del World Wide Web

Le pagine Web di un certo topic in genere possiedono link ad altre dello stesso topic (Linkage o Topical Locality).

Nella Web Social Network Analysis oltre al contenuto testuale delle risorse si sfruttano anche le informazioni contenute negli hyper-links.

I successivi algoritmi posso essere usati per diversi scopi:

- re-ranking dei risultati di un motore di ricerca in base alle informazioni estratte dai links
- analizzare la struttura e la dinamica del Web
- indirizzare il crawling dei motori di ricerca verso le risorse più interessanti (e.g. più alto rank)

3.2.1 HITS

Un link può essere letto come una indicazione di "autorevolezza" che chi l'ha creato (autore della pagina) vuole dare alla pagina puntata. Si potrebbe perciò pensare di aumentare l'importanza delle pagine con molti link entranti (back-link count).

Ma così ci limitiamo a considerare solo la "popolarità" assoluta di una pagina senza metterla in relazione con un certo argomento e con la "qualità" della pagina genitore.

HITS si basa sulla relazione tra pagine autorevoli (authoritative pages) per un certo topic, e pagine che puntano a molte pagine autorevoli (hubs). Se molte pagine di geocities puntano a java.sun.com, allora le pagine di geocities sono hub, mentre java.sun.com è una pagina autorevole.

Una pagina è ritenuta "importante" (alta authority) se riceve molti link da pagine importanti (con alta hubness). Di conseguenza e pagine hubs hanno la caratteristica di puntare a molte pagine importanti. Hubness e authority sono 2 misure correlate che vengono calcolate una in funzione dell'altra:

$$authority(p) = \sum_{q|q \to p} hub(q)$$

$$hub(p) = \sum_{q|p \to q} authority(q)$$

L'insieme di pagine viene ordinato per authority. Le hubs sono validi punti di partenza per esplorazioni.

3.2.2 PageRank

Ad ogni pagina si assegna una singola misura (rank) Una pagina ha alto rank se è alto il rank delle pagine che la puntano:

$$rank(p) = (1-d) \cdot \sum_{i=1}^{k} \frac{rank(p_i)}{C(p_i)} + d \cdot \frac{1}{T}$$

dove $C(p)$ è il numero di link dentro la pagina p, T è il totale di pagine e d è una costante $(0.1 \ldots 0.15)$.

Il rank può essere visto come la probabilità che si selezioni la pagina p. La d indica la probabilità che l'utente selezioni un'altra pagina.

3.2.3 HyperInformation

Una pagina ipertestuale ha un valore che non dipende solo dal suo contenuto testuale, ma anche dai link ivi contenuti. Se vi trovate in una pagina da cui potete raggiungerne altre pagine di vs interesse, la pagina è molto importante. Quando occorre valutare una pagina, si considera anche la misura di HyperInformation che tiene conto della presenza dei link:

$$Information(A) = TextInfo(A) + HyperInfo(A)$$

$$HyperInfo(A) = F^1 \cdot TextInfo(B_1) + F^2 \cdot TextInfo(B_2)$$

con $0 < F < 1$ la frazione della informazione testuale raggiungibile.

3.3 Focused Crawlers

3.3.1 Chakrabarti's Focused Crawler

Il Chakrabarti's Focused Crawler è un sistema di crawling autonomo per la ricerca di risorse inerenti un certo topic (rappresentato da un set di pagine iniziali fornite dall'utente).

È composto da 2 sotto-sistemi:

Classifier determina la rilevanza dei documenti rispetto al topic di interesse

Sfrutta una tassonomia gerarchica (Yahoo!) di categorie C per individuare gli argomenti che più interessano l'utente

1. L'utente suggerisce le pagine di interesse
2. Il sistema propone le categorie $C^* \subset C$ più affini con algoritmi di matching testuale (e.g. Vector space model)
3. L'utente eventualmente raffina le categorie (scegliendo quelle più generali o particolari)
4. Le classi finali vengono marcate come *good*.

Durante il crawling, ad ogni pagina che si visita viene associata la categoria più specifica c. Se uno dei nodi nei genitori di c è marcato come good, allora la pagina *non* viene ignorata. Ad esempio, se mi interessano le GT, e se il crawler ha trovato una pagina sulla Maserati, la considero buona comunque perché è "contenuta" in GT.

Distiller identifica i link che devono essere visitati per primi dal crawler.

Sfrutta HITS per individuare le risorse più importanti. Periodicamente viene eseguito l'algoritmo per individuare le pagine recuperate con più alto hubs, dopodiché si estraggono i link ivi contenuti, e si inseriscono nella coda delle risorse da visitare del crawler.

3.3.2 IBM's Intelligent Crawler

Si adatta alle risorse visitate durante il crawling stimando se una pagina è interessante per mezzo di algoritmi di machine learning. Non necessita quindi del classificatore con la gerarchia come nel focused crawling

$P(C)$ è la prob che una pagina sia di interesse. E sono i fatti che conosciamo riguardanti le candidate urls (testo pagine che le puntano, urls, testo ancore, etc). Esempio: cerchiamo pagine su "Bach". 0.3% di pagine di interesse. Ma se la parola "eshop" compare in una pagina che ha un link verso quella di interesse, la probabilità aumenta fino al 10%. La conoscenza E può aumentare la prob che una candidate url soddisfi il predicato.

$$P(C|E) > P(C)$$

nell'esempio infatti risulta $0.1 > 0.003$

3.3.3 Agent-based Focused Crawling

Sistemi di ricerca adattativi, che prendono spunto dal paradigma di programmazione Ant System:

> Come riescono gli animali a trovare il cammino e a coordinarsi? Quando una formica trova cibo, lascia delle tracce di feromone per marcare il percorso, in modo tale che le altre formiche possano trovarlo.

Si ha un'architettura reattiva formata da un numerosi ant-agent che vagano in ambiente di risorse ipertestuali. Ogni agente ha semplici comportamenti di basso livello che reagiscono a cambiamenti nell'ambiente. Le informazioni disponibili per un agente sono:

1. il risultato tra la query dell'utente e la risorsa corrente
2. l'intensità dei valori sui cammini di feromone, corrispondenti ai link uscenti

L'esecuzione del sistema si divide in cicli:

1. in ogni ciclo gli agenti compiono una sequenza di mosse da una risorsa a un'altra
2. alla fine del ciclo, ogni agente aggiorna le tracce di feromone del percorso effettuato come funzione dei punteggi pervenuti sulle risorse.
3. ad ogni nuovo ciclo, viene posizionato l'agente in una delle risorse iniziali
4. se una traccia esiste, l'agente decide di seguirla con una probabilità funzione della rispettiva intensità di feromone
5. se non esiste alcuna traccia, l'agente si muove casualmente

Se due percorsi portano a una pagina interessante, i primi agenti che raggiungono quella pagina sono gli unici che hanno seguito il percorso più corto, e quindi sono i primi a rilasciare il feromone che attrae gli agenti successivi allo stesso percorso.

Il sistema mostra due forme di adattività:

1. raffinamenti della query utente durante l'esecuzione
2. alterazione del contenuto delle risorse

Ad ogni ciclo le intensità di feromone sono aggiornate a seconda dei punteggi delle risorse visitate.

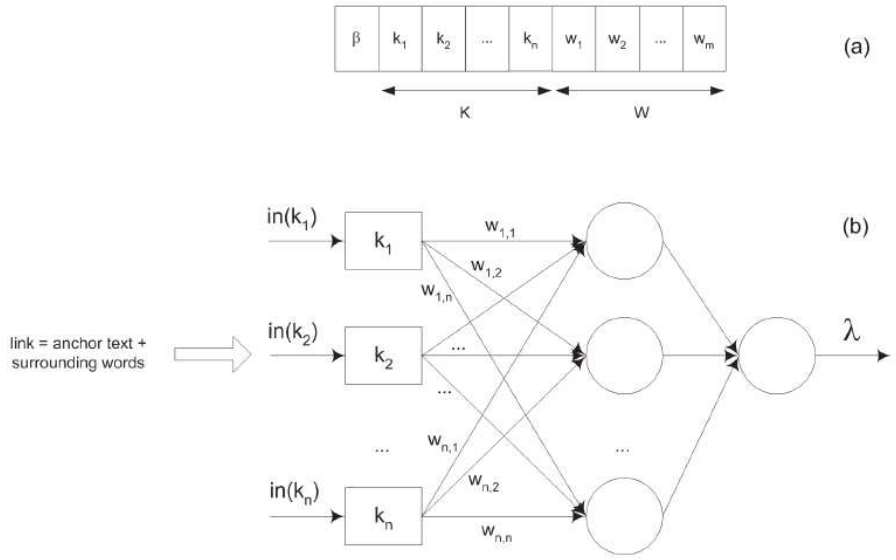

Figura 1: Agente nel genetic focused crawling

3.4 Genetic-based Focused Crawling

Una popolazione crescente di agenti intelligenti esplorano il Web guidati dalle query utente. Una popolazione di cromosomi codificati da una particolare struttura dati evolvono verso una soluzione potenziale attraverso un insieme di operatori generici. I cromosomi che arrivano più vicini alla soluzione migliore hanno maggiori probabilità di vivere e riprodursi.

Lo scopo è di imitare l'esplorazione umana con interazione bassa o nulla.

I genotipi (1) sono un insieme di cromosomi che determinano il comportamento di ricerca degli agenti. È formato da:

- un insieme di parole chiave K inizializzate con i termini di query.
- un vettore di pesi W, corrispondenti all'informazione memorizzata in una rete neurale, utilizzata per giudicare quali parole chiave nel primo insieme discrimano meglio i documenti rilevanti all'utente

4 Computer Vision

La Visione Artificiale (o Computer Vision o Pattern Recognition) si occupa dell'analisi e dell'interpretazione delle immagini digitali (per "permettere ad un computer di capire cosa sta guardando").

Le applicazioni pratiche sono il recupero d'immagini in database visivi (Image Retrieval) data la crescente diffusione di archivi d'informazione esclusivamente visiva (musei, archivi fotografici, e-commerce...). Visione applicata alla robotica. Sorveglianza automatica (tramite telecamere o altri dispositivi). Guida automatica di veicoli su strada. Visione industriale, medica, aerea, ...

Alcuni sottocampi:

low-level vision a partire da un'immagine I viene prodotta una seconda immagine $M(I)$ data dall'applicazione di filtri puntuali e/o locali. Si tratta quindi di una trasformazione dell'immagine.

medium-level vision estrazione di predeterminate caratteristiche dall'immagine. Dall'immagine I (o da $M(I)$) si passa ad un insieme di caratteristiche $F = \{f_1, \ldots, f_n\}$

high-level vision interpretazione dell'immagine (quali oggetti sono presenti o quali relazioni intercorrono tra essi)

La conoscenza del sistema generalmente è di tipo modellistico (model based) oppure appresa attraverso tecniche di machine learning.

I problemi principali nel riconoscimento di immagini:

Condizioni di illuminazione che producono una variazione nella distribuzione dell'intensità luminosa della scena.

Trasformazioni geometriche rigide dell'oggetto (in ordine di difficolta crescente):

1. roto-traslazioni e scalamenti in 2D,

2. roto-traslazioni e scalamenti in 3D.

Rumore

Gap tipo particolare di rumore consistente nella mancanza di elementi nell'immagine.

Occlusione

Segmentazione partizionamento dei dati di input in entita semantiche distinte (linee, regioni, oggetti).

Indexing effettuare una ricerca efficiente in un catalogo di modelli.

Identificazione riconoscere l'istanza di un oggetto in un'immagine.

Oggetti non rigidi (forbici, volti umani, ...). Il loro riconoscimento è complicato dalla possibilita che ha la loro forma di variare.

Classificazione riconoscere l'appartenenza ad una data classe di un oggetto in un'immagine.

4.1 Il problema della segmentazione

Col termine "segmentazione" in Computer Vision si intende un qualsiasi partizionamento dell'immagine (o della sequenza video) in insiemi omogenei rispetto ad una qualche caratteristica percettiva.

E importante sottolineare la trasversalita di tale definizione, ovvero il fatto che essa puo essere (ed in effetti è) applicata a vari livelli di processamento.

Nella low-level vision, con segmentazione si intendono quei processi data-driven (dipendenti solo dai dati) che permettono, ad esempio, di partizionare un'immagine in base al colore o un video in base alla luminosita della scena

In high-level vision, la segmentazione e uno dei problemi finora insormontabili per un riconoscimento effettivo degli oggetti rappresentati in un'immagine poiché è necessario che il sistema capisca che ci sono N oggetti distinti. Un oggetto, in sostanza, e un concetto "semantico" e non una primitiva percettiva come il colore e il suo riconoscimento non puo che dipendere dai modelli di oggetti in memoria (oltre che dai dati in input).

Vari studi psicologici sembrano confermare l'idea che, nel sistema visivo umano, il riconoscimento e la segmentazione non sono fasi nettamente distinte. La segmentazione umana sembra essere un processo sia data-driven che model-driven, in cui i risultati di una fase influiscono sulla fase successiva e viceversa.

Questo campo e ancora ampiamente inesplorato, sia a livello psicologico (si ricorda che i primi tentativi in tal senso risalgono alla teoria tedesca della Gestalt, negli anni '30), sia a livello computazionale.

4.2 Approccio statistico

Si scelgono una serie di n caratteristiche (features) dell'apparenza di un oggetto che siano facilmente misurabili. Il riconoscimento di O (con vettore $V(O)$) come istanza del modello M dipende dalla funzione: $dist(V(O), V(M)$.

Esempi tipici di features:

area della figura cioe il numero dei suoi pixel.

compattezza della regione definita come il rapporto tra il quadrato del perimetro e l'area.

allungamento della regione definito come il rapporto tra la lunghezza della corda di lunghezza massima e la lunghezza della corda ad essa perpendicolare.

minimo o massimo rettangolo (o ellissi o cerchio) incluso o includente la figura.

Medial Axis Transform di una figura piana, che ne restituisce uno scheletro.

Freeman Chain Code è possibile numerare convenzionalmente gli 8 (o 4) pixel confinanti col pixel p con valori in $[0, 7]$ e codificare con una stringa la sagoma di un oggetto.

coefficienti dell'espansione in serie di Fourier ottenuti interpretando la sagoma come fosse una funzione sinusoidale.

momenti digitali

centro di gravità o centroide

Vantaggi e svantaggi:

\+ Ottimo per l'indexing.

+ Permette di utilizzare tecniche di Machine Learning (Reti Neurali, metodi statistici...).

- Scarso potere discriminante.

- Non permette che l'oggetto d'interesse sia occluso o in contatto con altri oggetti nella scena.

4.3 Approccio strutturale (o tramite descrizioni relazionali)

Si scelgono una serie di primitive geometriche atomiche: segmenti, rettangoli, parallelepipedi o figure piu complesse, non necessariamente regolari.

Un oggetto e descritto tramite la combinazione delle primitive atomiche ottenuta mediante relazioni di tipo geometrico, gerarchico, ...

Vantaggi e svantaggi:

+ Permette una buona astrazione perché sfrutta il potere descrittivo tipico delle rappresentazioni simboliche.

- Spesso ha costi di unificazione esponenziali.

- Le descrizioni solitamente non sono precise. Ad esempio una casa può essere rappresentata come "una piramide sopra un parallelepipedo", ma questa piramide può essere più grande, spostata, ...

4.3.1 Attributed Relational Graph

Una struttura dati frequentemente usata, nel caso dell'approccio relazionale, e il grafo, che permette di associare ai nodi gli elementi atomici della descrizione, e agli archi gli elementi della relazione (o delle relazioni).

Ad esempio, attributi per i nodi possono essere l'area o la compattezza delle primitive geometriche ad essi associate. ' Un attributo per una relazione del tipo X e sopra Y potrebbe essere la misura della distanza fra gli elementi atomici X e Y Si parla, in tal caso, di *Attributed Relational Graph*.

Il Connection Graph, ad esempio ha:

Primitive regioni dell'immagine omogenee rispetto ai livelli di grigio

Unica relazione adiacenza tra regioni

Attributi dei nodi allungamento della regione

4.3.2 Syntactic Pattern Recognition

L'idea, non priva di un certo fascino, consiste nel considerare le immagini come fossero un testo da interpretare, descrivendo i modelli di interpretazione (pattern) mediante grammatiche formali e utilizzando, come procedura di matching, parser simili a quelli usati per l'interpretazione dei linguaggi formali.

Una chain code puo essere vista come un linguaggio molto semplice, contenente un'unica stringa.

4.3.3 Unificazione Mediante Grafi

Quando la struttura dati scelta per la rappresentazione di M e di I e il grafo, il cammino all'interno dell'albero di unificazione corrisponde all'implementazione di un **graph matching**.

L'isomorfismo tra grafi e un problema NP. Piu interessante, in Computer Vision, sono i problemi di:

- Isomorfismo tra sottografi (NP-Completo)
- Isomorfismo inesatto, in cui la funzione f e parziale e/o i vincoli sono solo parzialmente verificati (NP).

4.4 Approccio mediante allineamento

Usa rappresentazioni realistiche sia dell'oggetto O da riconoscere che dei modelli M in memoria. In fase di riconoscimento, formula una serie di ipotesi consistenti in trasformazioni geometriche T da applicare ad una delle due descrizioni (solitamente M) per tentare di farla "allineare" con l'altra. Al passo i-esimo l'ipotesi T_i viene verificata tramite misure di somiglianza (tipo l'overlapping tra i pixel).

4.4.1 Template Matching

Il template matching è il piu semplice algoritmo di allineamento. Esso consiste nel sovrapporre all'immagine in input I la sagoma M da riconoscere (modello 2D), traslando ed eventualmente ruotando M ad ogni iterazione.

Si tratta di una semplificazione dell'allineamento semplice gia incontrato, generalmente non considerando, per motivi di efficienza, la possibilita di ruotare il modello e restringendosi a template con sagome semplici.

E utilizzato in alcuni sistemi commerciali per interpretazioni di immagini industriali o mediche, nell'OCR, ...

La trasformata di Hough (1962) La Trasformata di Hough (HT) e una forma più elegante (e spesso più efficiente) di template matching fatta nello spazio dei parametri anziché nello spazio cartesiano.

Supponiamo di avere un'immagine I digitale descritta dalla funzione binaria f con valori in $\{0.1\}$. Supponiamo di cercare in essa la presenza di template descrivibili tramite semplici formule analitiche: ad esempio delle rette.

Se il punto p ($p = (x, y)$) è tale che $f(p) = 1$, allora tutte le retti passanti per p devono soddisfare:

$$y = mx + c$$

con m e c variabili. Per ogni p tale che $f(p) = 1$, l'algoritmo di Hough incrementa di un'unità l'accumulatore $H(m, c)$ ($H(m, c) = H(m, c) + 1$), per ogni c ed m tali che

$$c = -mx + y$$

Finita la scansione di I, i valori più alti in H indicheranno le rette con un maggior numero di pixel allineati.

I parametri c ed m sono difficilmente limitabili, percio normalmente si adotta una parametrizzazione diversa per le rette. Una retta r puo essere infatti espressa anche mediante la seguente formula:

$$\rho = \cos\theta x + \sin\theta y$$

dove ρ e θ sono, rispettivamente, la distanza e l'angolo formati dalla retta perpendicolre ad r e passante per l'origine.

Trasformata di Hough Generalizzata Ballard (1979-1981) propose una generalizzazione del metodo di Hough per trattare sagome qualsiasi, anche non regolari (Trasformata di Hough Generalizzata o GHT).

A run-time, per ogni pixel di I "voto" per la posizione del centro di massa.

Sia l'HT che la GHT sono metodi molto promettenti usati in letteratura in tantissime applicazioni diverse e con numerosissime varianti. Sono molto robusti a rumore ed occlusioni ma sono poco adatti per oggetti 3D. Infatti sono approcci viewer centered. Inoltre, risentono dei problemi generali dei metodi di allineamento ' (difficolta a trattare deformazioni).

4.4.2 L'algoritmo di Huttenlocher e Ullman (1990)

L'algoritmo di allineamento 3D proposto da Huttenlocher e Ullman parte dalla seguente considerazione geometrica:

> Dati tre punti non collineari m_1 , m_2 ed m_3 appartenenti al modello, e le loro rispettive proiezioni sul piano dell'immagine p_1 , p_2 e p_3 , esistono esattamente due trasformazioni (tra esse speculari) del modello tridimensionale nell'immagine bidimensionale.

In generale si tratta di un approccio preciso (non sempre veloce) che da buoni risultati nell'identificazione. Manca di capacita astrattive necessarie per la classificazione.

4.5 Applicazione della Computer Vision al recupero di immagini da database visivi

Un tipico sistema di recupero di immagini tramite somiglianza di forma presuppone:

- Una query by sketch: l'utente disegna una sagoma approssimativa di cio che sta cercando nel data base di immagini
- La ricerca nel data base avviene quindi tramite tecniche di indexing, identificazione e classificazione simili a quelle viste per il caso generale
- L'output del sistema non si limita a dire cosa e stato o non e stato riconosciuto, ma deve fornire una lista di immagini ordinate per grado di somiglianza con la query.

Si ritrovano i filoni classici, soprattutto quello statistico e l'allineamento (gli approcci strutturali qui sono meno comuni). L'allineamento, generalmente, avviene tramite trasformazioni geometriche non rigide T dello sketch iniziale per adattarlo come se fosse un "elastico" alle silhouette delle immagini in memoria.

4.5.1 Template matching elastico (deformabile)

Le immagini vengono inserite nel data base del sistema nella forma contenente solo gli edge. L'utente disegna il suo sketch usando un tool grafico e la sagoma finale viene rappresentata con una spline codificata mediante i suoi punti di controllo: $P = (p_1, \ldots, p_n)$

Se il matching tra lo sketch e l'immagine candidata e elevato, la procedura termina qui. Altrimenti, i vari p_i vengono "perturbati" in modo da modificare lo sketch e re-iterare il grado di correlazione.

E possibile massimizzare il grado di matching tra lo sketch e l'immagine M tramite un algoritmo iterativo che modifica progressivamente i p_i nella direzione di crescita massima ottenuta derivando M rispetto a P (tecnica del gradiente ascendente).

5 Clustering – Metriche e distanze

5.1 Data mining

Il data mining è il processo di analisi, svolto in modo semiautomatico, di una grande quantità di dati grezzi al fine di scoprire il modello ("pattern") che li governa, o una regola significativa, da cui ricavare conoscenze utili applicabili al nostro contesto operativo, come ad esempio:

Previsioni identificano relazioni e tendenze nei dati, aiutando a scoprire fenomeni di mercato

Verifiche servono a convalidare le scoperte fatte in fase di analisi per garantire decisioni corrette

> Il Data Mining è la risposta tecnologica all'esigenza di saper analizzare e ricavare conoscenze utili, dalle enormi quantità di dati grezzi che si raccolgono ormai in tutti i contesti operativi della nostra società: **estrazione delle informazioni implicite**.

Vengono utilizzati come tipi di dati:

Matrice dei dati (n oggetti con p attributi):

$$\begin{pmatrix} x_{11} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{np} \end{pmatrix}$$

Matrice di dissimilarità dove $d(i, j)$ è la misura di dissimilarità tra oggetti i e j. Se $d(i, j) = 0$ allora gli oggetti sono molto simili. Ad esempio, un giocattolo con tre oggetti:

$$\begin{pmatrix} 0 & \cdots & \cdots \\ d(2,1) & 0 & \cdots \\ d(3,1) & d(3,2) & 0 \end{pmatrix}$$

Variabili numeriche, binarie, categoriche nominali, di tipo misto

5.2 La distanza

Proprietà della distanza (dalla matrice di dissimilarità):

$$d(a, b) \geq 0 \tag{1}$$
$$d(a, b) = d(b, a) \tag{2}$$
$$d(a, a) = 0 \tag{3}$$
$$d(a, b) \geq 0 \tag{4}$$
$$d(a, b) \leq d(a, c) + d(c, b) \tag{5}$$
$$\tag{6}$$

Per misurare similarità tra coppie di oggetti spesso si utilizza la **distanza di MINKOWSKI**:

$$d(i, j) = \sqrt[q]{|x_{i1} - x_{j1}|^q + \ldots + |x_{ip} - x_{jp}|^q}$$

dove q è un intero positivo:

- se $q = 1$ si ha la distanza di Manhattan
- se $q = 2$ si ha la distanza euclidea

Possiamo pesare le variabili, ottenendo una misura di distanza pesata.

5.3 Partitioning Method

Scopo: partizionare il database D di n oggetti in un insieme di k cluster.

5.3.1 Apprendimento non supervisionato

K-MEANS

1. scegliamo $k = 3$ e i tre semi iniziali
2. assegniamo ogni record al cluster con il centroide (o seme) più vicino
3. il passo due ha individuato nuovi cluster. Ci calcoliamo i centroidi (o semi) di questi

La complessità è $O(nktd)$, dove:

n numero di oggetti

k numero di cluster

t numero di iterazioni

d numero di attributi

K-MEDIOIDS (PAM)

5.3.2 Apprendimento supervisionato

Reti neurali (MLP) Una *Multi-Layers Perceptron net* (*MLP*) è composta da una serie di "percettroni" organizzati con una struttura gerarchica che può comprendere uno o più strati nascosti (hidden layer) legati con la regola del feed-forward (un nodo dello strato i-esimo può essere collegato solo ad un nodo dello strato $i + 1$-esimo).

Il *percettrone* è un semplice neurone dotato del circuito "teacher" per l'apprendimento:

Quando usare le reti neurali nel data mining ?

Le reti neurali rappresentano una buona scelta in caso di classificazioni e previsioni, quando è più importante avere velocemente e bene i risultati di un modello che non sapere come questo funziona.

Le reti neurali non funzionano bene quando si ha a che fare con centinaia o migliaia di variabili di input. Un gran numero di queste caratteristiche rende più difficile il compito della rete di individuare pattern e la fase di training può prolungarsi nel tempo senza trovare una buona soluzione. In questo caso, le reti neurali danno frutti migliori se combinate con gli alberi decisionali: questi ultimi infatti selezionano le variabili più importanti, impiegate poi per il training della rete

6 Machine Learning

6.1 Problemi di apprendimento Well-Posed

> A *computer program* is said *to learn* from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T , as measured by P, improves with experience E

Concetti fondamentali:

Task T obiettivo del sistema (giocare a dama)

Experience E insieme di addestramento dal quale apprendere (partite giocate)

Performance measure P misura della capacità di eseguire il task (# di partite vinte)

6.1.1 Esempi di problemi well-posed

Gioco della dama:

Task T giocare a dama

Experience E partite giocate contro se stesso

Performance measure P % di partite vinte

Riconoscimento caratteri scritti a mano

Task T riconoscere e classificare parole scritte a mano memorizzate come immagini

Experience E un database di parole scritte a mano insieme alla loro classificazione

Performance measure P % di parole correttamente classificate

Robot che guida un autoveicolo

Task T guidare un autoveicolo su una strada pubblica

Experience E sequenza di immagini

Performance measure P distanza percorsa prima di un errore

6.2 Progettazione di un sistema di apprendimento

Gli obiettivi generali sono:

- Definire in modo preciso una classe generale di problemi che includono forme interessanti di apprendimento
- Esplorare algoritmi che risolvono problemi di apprendimento
- Comprendere la struttura fondamentale di problemi e processi di apprendimento

6.2.1 Scelta della Training Experience

La Training Experience è caratterizzata da:

Feedback **Direct feedback** training examples: insieme di singole configurazioni della scacchiera insieme alla mossa corretta per ciascuno di essi.

Indirect feedback training examples: insieme di sequenze di mosse insieme ai risultati finali delle partite giocate (è difficile ricavare informazioni sulla bontà delle singole mosse)

Controllo della sequenza di training examples

Distribuzione dei training examples: quanto rappresenta bene la distribuzione di esempi sui quali il sistema verrà misurato

Come assunzione, il sistema apprende attraverso partite contro sè stesso. Non necessita di un external trainer e possono essere generati molti training esempi

6.2.2 Case study: il gioco della dama

Da stabilire:

1. Il tipo esatto di conoscenza da apprendere
2. Una rappresentazione per la conoscenza target
3. Un meccanismo di apprendimento

La scelta della mossa è però difficile da apprendere per via del tipo di indirect experience disponibile al sistema. Nuova Target Function V: mapping tra ogni stato ammesso della scacchiera e l'insieme dei numeri reali: punteggio più alto allo stato più promettente per l'esito della partita. Definiamo noi una funzione (ricorsiva) $V(b)$, dove b è lo stato scacchiera, $b \in B$:

- Se b è uno stato finale positivo $V(b) = 100$
- Se b è uno stato finale negativo $V(b) = -100$
- Se b è uno stato finale "patta" $V(b) = 0$
- Se b è uno stato intermedio $V(b) = V(b')$ essendo b' il migliore stato finale che può essere raggiunto a partire dallo stato b e giocando in modo ottimo fino alla fine del gioco.

E' pero una definizione nonoperazionale: V non può essere realmente computata. Si dice definizione **operazionale** di V una definizione che può essere **realmente** utilizzata dal sistema per valutare stati e selezionare le giuste mosse in limiti di tempo realistici

Si approssima la funzione target ideale V con una funzione V^*:

- X_1: numero di pedine nere sulla scacchiera
- X_2: numero di pedine bianche sulla scacchiera
- X_3: numero di dame nere sulla scacchiera
- X_4: numero di dame bianche sulla scacchiera
- X_5: numero di pedine nere mangiate dal bianco
- X_6: numero di pedine bianche mangiate dal nero

$$V^*(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$$

Esempio di training: coppia ordinata della forma $< b, V_{train}(b) >$ essendo b lo stato della scacchiera e $V_{train}(b)$ il valore associato ad esso. Esempio di training: $< 3, 0, 1, 0, 0, 0, +100 >$: il bianco ha vinto.

Gli esempi di training disponibili per l'apprendimento sono però di tipo indiretto: le informazioni disponibili sono solo vittoria o sconfitta (quindi di fine partita). Bisogna:

- Costruire una procedura che determini gli esempi di training a partire dall'esperienza indiretta disponibile al learner
- Modificare i pesi w in modo tale da approssimare al meglio la funzione target V

Si ha bisogno quindi di esempi che assegnino agli stati della scacchiera un punteggio Per gli stati finali è facile mentre per gli stati intermedi? Poniamo il valore associato a ciascuno stato intermedio uguale al valore dello stato successivo:

Bisogna determinare i pesi w in modo tale da adattare al massimo (*best fit*) l'algoritmo agli esempi di addestramento $< b, V_{train}(b) >$ Regola *Least Mean Squares* (LMS): si definisce la migliore hypothesis o insieme dei pesi come quella/o che minimizza l'errore quadratico E: $(\sum (v_{train}(b) - v^*(b))^2)$

Regola LMS per l'aggiornamento dei pesi w: Per ogni training example $< b, V_{train}(b) >$ Calcola con i pesi attuali V^*. Aggiorna ciascun peso w_i con la regola

$$w_i = w_i + \eta(v_{train}(b) - v^*(b))x_i$$

dove η è una costante $(0, 1)$ che modera la velocità di aggiornamento dei pesi w.

6.2.3 Schema di un sistema generale di apprendimento

Il progetto completo di un Learning System prevede:

modulo Performance System è il modulo che gioca a dama utilizzando la V^* La sua performance migliora con il numero di partite giocate

modulo Critic prende in input una partita e produce esempi di training

modulo Generalizer implementa la LMS rule modificando i pesi w (generalizza sempre di più l'ipotesi V^*)

modulo Experiment Generator prende in input la ipotesi V^* corrente e genera un nuovo problema da esplorare (una nuova partita)

6.3 Concept Learning

6.3.1 Definizioni

Learning indurre funzioni generali da esempi particolari di training

Concept learning acquisire la definizione di una categoria generale dato un insieme di esempi positivi e negativi della categoria stessa

Best Fit hypothesis determinare l'ipotesi h che meglio si adatta agli esempi di training

General-to-Specific introduzione nello spazio delle ipotesi H di un ordinamento parziale

Find-s/Version Space studio di algoritmi che convergono in H all'ipotesi h corretta.

Nel Concept Learning si tratta di

> Inferenziare automaticamente una funzione Learning booleana a partire dall'insieme dei training examples
>
> $$< input, output >$$

6.3.2 Un esempio di concept learning

Ad esempio, se il Target Concept è "Giorni in cui il mio amico Aldo pratica il suo sport d'acqua preferito", dato un training set di input, il Concept Learning porta ad apprendere a predire il valore di EnjoySport per un giorno *arbitrario*, sulla base degli attributi del giorno stesso.

6.3.3 Concept Learning come ricerca in H

Per la rappresentazione della funzione ipotesi h, nell'ipotesi semplice, ogni ipotesi h consiste nell'unione dei vincoli espressi negli attributi ammessi nella singola istanza. Possibile rappresentazione:

- Un vettore di 6 elementi/vincoli che specificano i valori dei 6 attributi Sky, AirTemp, Humidity, Wind, Water e Forecast
- Per ogni attributo, l'ipotesi potrà contenere i seguenti valori:
  - ? → ogni valore dell'attributo è accettabile
  - Il valore tra quelli consentiti
  - 0 → nessun valore per quell'attributo
- Se una istanza x soddisfa tutti i vincoli dell'ipotesi h allora h classifica x come esempio positivo: $h(x) = 1$

Es. di ipotesi h: $<?, Cold, High, ?, ?, ?>$. Aldo pratica lo sport solo nei giorni freddi con umidità alta. Ipotesi h più generale $<?, ?, ?, ?, ?, ?>$: ogni giorno è buono per fare sport d'acqua Ipotesi h più specifica $< 0, 0, 0, 0, 0, 0 >$: nessun giorno è buono per fare sport d'acqua
Definizioni:

Insieme delle istanze X l'insieme di items sui quali il concetto è definito. (Insieme di tutti i possibili giorni)

Target Concept c la funzione/concept che deve essere appresa. In generale c può essere qualunque funzione booleana definita su X: c: X->[0,1]

Training Examples D : formato da n istanze $x \in X$ insieme a n valori corrispondenti c(x): <x, c(x)>. Se c(x)=1 si parla di esempi positivi mentre per c(x)=0 si parla di esempi negativi

Insieme H insieme di tutte le possibili ipotesi h candidate alla determinazione della funzione target

Obiettivo determinare $h \in H$ tale che $h(x) = c(x)$ per tutti gli $x \in X$

Dati:

Istanze X giorni possibili, ciascuno dei quali descritto dagli attributi: Sky (Sunny, Cloudy e Rainy), AirTemp (Warm, Cold), ...

Ipotesi H ciascuna ipotesi h è descritta dall'unione di vincoli sugli attributi 1,...,6 + <?> e <0>

Concetto Target C EnjoySport: X->0,1

Esempi di training D esempi positivi e negativi della funzione target

Determinare una ipotesi $h \in H$ tale che h(x)=c(x) $\forall x \in X$.

6.3.4 Concept Learning as Search

Il concept learning può essere visto come il compito di trovare la funzione h attraverso una ricerca nello spazio H Obiettivo della ricerca in H: trovare la funzione h che meglio approssimi c sui training examples (best fit h). È necessario dunque lo studio di algoritmi e tecniche di ricerca in H.

Si introduce nell'insieme H un ordinamento che faciliti la ricerca della funzione h. Ogni istanza classificata come positiva da h_1 è classificata come positiva anche da h_2, cioè h_2 più generale di h_1. L'ordinamento è quindi dal più specifico al più generale. L'ordinamento parziale $\geq_g$ proposto gode della proprietà di riflessività, di antisimmetria e di transitività.

6.3.5 Algoritmo Find-S

Consente la ricerca dell'ipotesi più specifica:

1. Inizializza h alla ipotesi più specifica in H.
2. Per ogni istanza positiva di training x e per ogni attributo con vincolo a_i in h, se il vincolo a_i non è soddisfatto da x allora sostituisci a_i in h con il vincolo più generale soddisfatto da x

È quindi più specifica rispetto a tutte le ipotesi fornite.

h di Find-S è il target concept? Find-S non ci garantisce che non esistano altre ipotesi h in H consistenti con i dati. Ci vorrebbe un algoritmo che desse maggiori informazioni sullo spazio H e h. Perchè predilire l'ipotesi più specifica? Quanto sono consistenti gli esempi di training? Find-S può essere guidato male dal rumore(ignorando esempi negativi). Vorremmo un algoritmo che tenesse conto del rumore nei dati Se esistono più ipotesi h consistenti con i dati?

La risposta sta nel Version Space ed altri algoritmi.

6.3.6 Version Space

Una ipotesi h è consistente con un insieme di training examples D se e solo se h(x)=c(x) per ogni esempio <x, c(x)> in D.

Il version space è il sottoinsieme di H formato da tutte le ipotesi h consistenti in D

L'algoritmo List-Then-Eliminate fornisce il set di ipotesi consistente con il training set ed appartenente al training set.

Version Space: lo rappresentiamo con i suoi membri più generali e meno generali. Essi formano due insiemi di confine, G ed S (General e Specific) che delimitano il version space dentro lo spazio parzialmente ordinato delle ipotesi.

Algoritmo di apprendimento Candidate-Elimination Calcola il Version Space contenente tutte le ipotesi di H che sono consistenti con una sequenza osservata di esempi di training. Parte inizializzando il VS all'insieme di tutte le ipotesi in H, ossia inizializzando G ed S. G_0 {<?, ?, ?, ?, ?, ?>} S_0 {<0, 0, 0, 0, 0, 0>} L'algoritmo ritorna il VS contenente tutte le ipotesi consistenti con gli esempi e solo quelle.

Converge all'ipotesi h corretta a patto che non ci siano errori negli errori di training e esista in H l'ipotesi h. Può convergere anche ad un insieme vuoto.

7 Logica Fuzzy

Le tecniche di controllo fuzzy sono oggi estensivamente utilizzate in molti settori; tra questi si citano apparecchiature di largo consumo come lavatrici e telecamere ma anche impianti di depurazione delle acque e cambi automatici di vetture di prestigio. Alla base di questo approccio è in genere la realizzazione di un sistema di controllo che incorpora e emula, tramite regole, la conoscenza di esperti del settore. Altre applicazioni sviluppate recentemente riguardano il trattamento di dati sensoriali per l'estrazione di informazioni in presenza di forti incertezze non strutturate.

Quando usare il controllo "intelligente" fuzzy? Quando i task sono ben posti, si hanno modelli affidabili e conoscenze quantitative si utilizzano *metodi di controllo classici*. Viceversa, quando i task sono complessi, modelli incerti e un elevato costo della conoscenza *quantitativa* si applica invece il *controllo intelligente*. All'interno di tale scelta, se la conoscenza *qualitativa* è disponibile ed è utile si usano *metodi "Rule based" o "algoritmici" (fuzzy)*. Se invece la conoscenza qualitativa non è sfruttabile, si adottano *approcci ad apprendimento (Learning)*, come le reti neurali.

7.1 Insiemi Fuzzy

Un elemento x dell'insieme universale U può appartenere o meno ad un insieme A. L'insieme A è definito dalla sua fcn. caratteristica:

$$\mu_A : U \to \{0, 1\} \tag{7}$$

$$\mu_A = A(x) = \begin{cases} 1 & \text{se } x \in A \\ 0 & \text{altrimenti} \end{cases} \tag{8}$$

Ho quindi solo valori discreti: 0 o 1.

Un *insieme fuzzy* è un'estensione per la quale la funzione caratteristica è continua.

Il significato è quello di graduare, sfumare l'appartenenza. Quanto vale per gli insiemi vale per le proposizioni, calcolando il grado di verità di un'affermazione. Se U è discreto, si può enumerare il fset.

Il problema di questo approccio è determinare qual è l'andamento giusto della funzione. La scelta della funzione caratteristica è legata infatti alla soggettività, all'arbitrarietà. La differenza significativa della logica fuzzy rispetto alle altre logiche si può notare con il seguente esempio. Se la domanda è: "C'è un panino col salame in frigo?" è la risposta ha valore 0.5, se la interpretiamo come:

probabilità significa "forse"

misura significa che c'è mezzo panino

fuzzy significa che c'è *qualcosa* (magari pane e salame separati, oppure un panino al prosciutto, ...)

Ora elenchiamo i concetti di base degli insiemi fuzzy $A(x)$:

supporto $x : A(x) > 0$

altezza $\max(A(x))$. Spesso si normalizza a 1.

cardinalità $card(A(x)) = \sum A(x_i)$. Vale per U discreto

α-cuts $A_\alpha = \{x \in X : A(x) > \alpha\}$. È un concetto utile per descrivere i fuzzy set e per comprendere alcune proprietà. È un insieme *crisp*. Nella definizione, α è variabile, e il valore A_α potrà essere vari insiemi di x "spezzati", tali per cui la proprietà vale.

7.1.1 Operatori logici

Per introdurli usiamo un'approccio assiomatico: sono funzioni che soddisfano assiomi "desiderabili". Assiomi per il complemento $c : [0,1] \to [0,1]$:

1. condizioni al contorno: $c(0) = 1$, $c(1) = 0$
2. monotona non decrescente: $a < b \to c(a) \geq c(b)$
3. c è una funzione continua (non indispensabile)
4. funzione involutiva: $c(c(x)) = x$ (non indispensabile)

Assiomi per l'unione $u : [0,1] \times [0,1] \to [0,1]$ (altra notazione: $\vee$):

1. condizioni al contorno: $u(0,0) = 0$; $u(0,1) = u(1,0) = u(1,1) = 1$
2. $u(a,b) = u(b,a)$
3. se $a < a'$ e $b < b'$, allora $u(a,b) < u(a',b')$
4. proprietà distributiva: $u(u(a,b),c) = u(a,u(b,c))$
5. u è una funzione continua (non indispensabile)
6. idempotenza: $u(a,a) = a$ (non indispensabile)

Assiomi per l'intersezione $i : [0,1] \times [0,1] \to [0,1]$ (altra notazione: $\wedge$):

1. condizioni al contorno: $i(1,1) = 1$; $i(0,1) = i(1,0) = i(0,0) = 0$
2. $i(a,b) = i(b,a)$
3. se $a < a'$ e $b < b'$, allora $i(a,b) < i(a',b')$
4. proprietà distributiva: $i(i(a,b),c) = i(a,i(b,c))$
5. i è una funzione continua (non indispensabile)
6. idempotenza: $i(a,a) = a$ (non indispensabile)

Le forme più impiegate sono:

$c(x)$	$u(a,b)$	$i(a,b)$	Nome u ed i
$1-x$	$\max(a,b)$	$\min(a,b)$	MaxMin (MM)
	$a+b-ab$	ab	PRod (PR)
	$\min(1,a+b)$	$1-\min(1,2-a-b)$	BS

Con le forme impiegate risultano ovviamente soddisfatte le leggi di De Morgan:

$$c[i(a,b)] = u[c(a), c(b)]$$

Risulta inoltre che, nel caso di unione u:

$$MM \subset PR \subset BS$$

e, viceversa, nel caso di intersezione i:

$$MM \supset PR \supset BS$$

da cui si deduce che MM è l'approccio più "discreto", mentre BS è l'approccio più "estremista". Tutte le forme individuano la più conveniente, ma solo PR contiene tutte le sfumature, poiché la funzione $\min(a,b)$ è non-interagente. L'approccio BS rifiuta tutto al di sotto di un limite.

A causa dell'idempotenza e della proprietà distributiva che caratterizzano u ed i, la perdita di informazioni è vista come un *arricchimento di informazioni*. Si ha quindi che $i(a, c(a)) \neq 0$ e $u(a, c(a)) \neq 1$. Infatti una proposizione può essere al tempo stesso un po' vera e un po' falsa (frase che troverebbe molti dubbi nella logica classica...).

7.1.2 Implicazioni

L'implicazione coincide con l'inclusione: l'essere sottoinsieme di. Risulta infatti che $B \Rightarrow A$ se $B \subset A$. Per esprimere l'inclusione si usano le seguenti espressioni a più valori:

Goedel $B(x) \subset A(x) = \begin{cases} 1 & \text{se } B(x) \leq A(x) \\ A(x) & \text{altrimenti} \end{cases}$

Goguen $B(x) \subset A(x) = \begin{cases} 1 & \text{se } B(x) = 0 \\ \min\left(1, \frac{A(x)}{B(x)}\right) & \text{altrimenti} \end{cases}$

Lukas $B(x) \subset A(x) = \min(1, 1 + A(x) - B(x))$

Quando $B(x) \leq A(x)$, tutte le 3 espressioni sono uguali a 1. Le differenze si vedono quando $B(x) > A(x)$. Con Goedel è uguale ad $A(x)$, con Goguen è uguale a $\frac{A(x)}{B(x)}$, con Lukas è uguale a $1 + A(x) - B(x)$. Praticamente, nell'osservare se A include B ($A \supset B$), risulta che, finquando $A \not\supset B$ con Goedel si segue l'andamento di A, con Goguen si segue si fa più o meno una media tra A e B, mentre con Lukas a distanze minori tra A e B coincide un valore maggiore (quindi ho valori elevati anche quando $A \not\supset B$ nel caso ad esempio che $A = 0$ e $B = 0.05$).

7.2 Relazioni

7.2.1 Prodotto cartesiano e relazioni

Dati due insiemi (crisp) A e B, il *prodotto cartesiano* è un insieme fuzzy multidimensionale:

$$Q = A \times B = \{(a,b) : a \in A, b \in B\}$$

La *relazione* R è un sottoinsieme di Q:

$$R(a,b) = \begin{cases} 1 & a, b \in R \\ 0 & \text{altrimenti} \end{cases}$$

La *proiezione proj* restituisce il valore più alto associato alla riga o colonna proiettata[1]:

$$P(x) = sup_y R(x,y)$$

La *estensione cilindrica cyl* copia il valore di una riga per il numero delle colonne y:

$$R(x,y) = F(x) \forall y$$

La *chiusura cilindrica* è la massima relazione compatibile con due proiezioni. Si ottiene come:

$$\wedge(proj's)$$

[1] si chiama proiezione proprio perché riduce i valori possibili da $N \times M$ a $N + M$

7.2.2 Composizione

Date due relazioni P e Q definite su $X \times Y$ e $Y \times Z$, la *composizione di relazioni* ne trova una terza T su $X \times Z$:

$$T = P \circ Q$$

con assiomi:

1. $(P \circ Q)^T = P^T \circ Q^T$
2. $(P \circ Q) \circ R = P \circ (Q \circ R)$
3. $P \circ Q = Q \circ T$ (non richiesto)

La composizione si realizza nel seguente modo:

$$T(x, z) = \text{proj}_{y \in Y}[R(x, y) \wedge Q(y, z)] \tag{9}$$

Se risulta $\wedge \rightarrow \min$ e $\text{proj} \rightarrow \max$, allora:

$$T(x, z) = \max_{y \in Y}[\min(R(x, y), Q(y, z))] \tag{10}$$

Ad esempio, per calcolare la composizione

1. si scelgono due valori x e z (ad esempio $x = Londra$ e $z = Roma$)
2. si calcola il minimo valore tra $R(x, y_1)$ e $Q(y_1, z)$. Ad esempio:

   $$lontanoDa(Londra, Parigi) = 0.8$$

   $$lontanoDa(Parigi, Roma) = 0.7$$

   allora restituisce $\min(0.8, 0.7) = 0.7$
3. si calcola il minimo valore tra $R(x, y_2)$ e $Q(y_2, z)$. Ad esempio:

   $$lontanoDa(Londra, NYC) = 0.4$$

   $$lontanoDa(NYC, Roma) = 0.2$$

   allora restituisce $\min(0.4, 0.2) = 0.2$
4. si calcola il massimo valore ottenuto: $\max(0.2, 0.7) = 0.7$, cioè:

   $$lontanoDa(Londra, Roma) = 0.7$$

7.3 Ragionamento fuzzy

7.3.1 Regole

Il ragionamento fuzzy è un sistema basato su regole. È formato da:

conoscenza attuale a' (composta da k fatti)

regole nella forma $a \Rightarrow b$ (anche nota come conoscenza a priori)

deduzione b'

Il passaggio regole/deduzione è possibile grazie al *modus ponens*:

$$a' \circ (a \Rightarrow b) =: b'$$

7.3.2 Inferenze

Le inferenze sono composte da:

antecedente velocità è xxx

regola IF "velocità è elevata" THEN "azione è frena"

conseguente azione è yyy

Le regole dovrebbero essere:

- complete: deve ricoprire a sufficienza le possibili situazioni
- non contraddittorie: regole con antecedenti simili o adiacenti non dovrebbero dare uscite molto diverse tra loro

Se la regola che conosciamo è

```
velocità alta --> frena
```

Ma allora è valido

```
velocità molto alta --> frena molto?
```

Secondo l'approccio logic-based non sappiamo cosa fare, ci vogliono altre regole. Il *modus ponens generalizzato* (GMP) ammette le estensioni richieste ed è molto usato:

- Consente di ridurre il numero di regole
- Sembra più aderente al ragionamento quotidiano

7.3.3 Approccio logic-based

Si basa sull'equivalenza:

$$A \Rightarrow B \equiv not(A) \vee B$$

È necessario risolvere tre problemi:

1. costruire la relazione R dalla regola in modo che soddisfi il modus ponens $a \circ (a \Rightarrow b) = b$
2. come gestire più regole
3. qual è il valore di b' se a' non è esattamente a

La soluzione viene dalla teoria delle equazioni fuzzy, imponendo che $b' = b$ quando $a' = a$ e cercando la soluzione massima (include le altre).

Se l'inferenza è MM, allora $R\ (A \Rightarrow B)$ va costruita come Goedel. Con questa combinazione è interessante notare che:

1. la massima indeterminatezza è $= 1$ e vale se:

$$A(x) = 0 \quad \forall x \qquad \Rightarrow R(x, y) = 1 \quad \forall x, y$$

2. se l'antecedente è più specifico, il conseguente mantiene il grado di incertezza iniziale:

$$A'(x) \subset A(x) \quad \forall x \qquad \Rightarrow b'(z) = b(z)$$

3. se l'antecedente non coincide, il conseguente è più incerto:

$$A'(x) \neq A(x) \qquad \Rightarrow b'(z) \supset b(z)$$

Per la composizione di più regole, abbiamo più regole con gli stessi antecedenti:

$$a' \circ (a_k \Rightarrow b_k) = b'_k$$

e dobbiamo ricavare il fuzzy set b'. Essendo la massima indeterminatezza pari a 1, si dovrà porre:

$$b'(z) = \min[b'_k(x)]$$

7.3.4 Approccio generalizzato

L'approccio GMP è piuttosto euristico e si basa più sulla "soddisfazione dell'utente" che su basi matematiche. Si cerca un legame "più forte" tra antecedenti e conseguente, quasi funzionale.

7.3.5 Impieghi nei controlli

Un sistema (ideale) di controllo fuzzy prevede:

fuzzify associa a ogni misura (proveniente dal processo) un fset (in base a precisione-rumore) che consente di descrivere in modo graduale una misura (ad esempio "temperatura alta").

motore inferenziale che opera su regole ricavate in base a interviste ad esperti o anche attraverso metodi adattativi, ad apprendimento, misti neurale-fuzzy.

de-fuzzify ricava dalle conseguenze (fuzzy) (ad esempio "velocità dell'aria"[2]) il valore più plausibile:

$$x_{CoG} = \frac{\sum A(x_i)x_i}{\sum A(x_i)}$$

processo riceve in input i controlli eseguiti dal de-fuzzify, da cui ricava delle misure, che vengono inviate al fuzzify

Le misure si assumono esatte, quindi singleton (fset con un solo elemento diverso da zero, pari a 1). L'incertezza è quindi solo nelle regole fornite dall'esperto.

Nell'*approccio TSK* gli antecedenti vengono composti con il *min* o il *pr*:

```
IF x IS a AND y IS b THEN z=px+qy+r
```

È un approccio più "controllistico" poiché si ha un controllore lineare per ogni condizione di funzionamento, con transizione graduale dall'uno all'altro. Ottimi risultati pratici ed implementazione efficiente. È meno linguistico e meno facile da ricavare tramite interviste. È più adatto a studi analitici, ottimizzazione, apprendimento.

In conclusione, quando usare il controllo fuzzy? In caso di ibridazione di sistemi differenziali, il sistema di controllo risulta non lineare e si hanno problemi con le specifiche del controllo fuzzy. D'altra parte, se il modello dinamico non è disponibile o è particolarmente complesso, si avrebbero le stesse difficoltà anche con gli approcci tradizionali. Sono questi i casi in cui il controllo fuzzy è una possibile strada.

7.4 Misure Fuzzy e Ragionamento con Incertezza

7.4.1 Tipologie di "ignoranza"

Sono classificabili le seguenti tipologie di "ignoranza":

incompletezza elementi non presenti, sensori guasti

imprecisione sensori imprecisi o rumorosi

incertezza da quale sensore viene l'informazione?

La teoria delle misure fuzzy può trattare le ultime due

7.4.2 Misure Fuzzy

Le misure fuzzy non hanno nulla di fuzzy, ma sono usate per il calcolo dell'incertezza. Dato l'insieme universale U e l'insieme vuoto $\oslash$, risulta:

1. $g(\oslash) = 0$; $g(U) = 1$
2. $A \subseteq B \Rightarrow g(A) \leq g(B)$
3. $\lim_{n\to\infty} g(A_n) = g(\lim_{n\to\infty} A_n)$

Le misure sono spesso definite sull'insieme delle parti di U: $\mathcal{P}(U)$.

[2] sapendo che la misura è "temperatura alta", la velocità dell'aria è una conseguenza a seconda di quanto la temperatura è alta

7.4.3 Misure di Credibilità-Plausibilità

Il significato è il seguente:

Plausibilità (plausibility, pl) misura delle "intensità" delle prove che non contrastano con un'ipotesi. Ad es. "il motore non parte". Che il problema sia la benzina o la batteria sono alternative entrambe plausibili (quindi risulta $\text{pl}(benzina) = \text{pl}(batteria) = 1$)

Credibilità (belief, bel) misura delle "intensità" delle prove a favore di un'ipotesi. Ad es. se la spia di riserva è accesa, aumenta la credibilità che il problema sia la benzina (aumenta $\text{bel}(benzina)$, diminuisce $\text{pl}(batteria)$)

Nella credibilità, oltre ai tre assiomi visti nel par. 7.4.2, risulta anche:

4. proprietà di superadditività per ogni n:

$$\text{bel}(A_1 \cup \ldots \cup A_n) \geq \sum_k \text{bel}(A_i) - \sum_{k<h} \text{bel}(A_k \cap A_h) + \ldots + (-1)^{n+1} \text{bel}(A_1 \cap \ldots \cap A_n)$$

per $n = 2$:

$$\text{bel}(A_1 \cup A_2) \geq \text{bel}(A_1) + \text{bel}(A_2) - \text{bel}(A_1 \cap A_2)$$

come dire: la credibilità che il problema della macchina siano la benzina o la batteria è la credibilità di ognuna, ma tolta la credibilità che il problema sia dovuto da entrambi contemporaneamente, che costituisce un'altro caso.

Dalla proprietà segue che:

$$\text{bel}\left(A \cup \overline{A}\right) = \text{bel}(U) = 1 \geq \text{bel}(A) + \text{bel}\left(\overline{A}\right)$$

Data una misura bel, la corrispondente misura di plausibilità pl è

$$\text{pl}(A) = 1 - \text{bel}\left(\overline{A}\right)$$

Nella plausibilità, oltre ai tre assiomi visti nel par. 7.4.2, risulta anche:

4. proprietà di subadditività per ogni n:

$$\text{pl}(A_1 \cap \ldots \cap A_n) \leq \sum_k \text{pl}(A_i) - \sum_{k<h} \text{pl}(A_k \cup A_h) + \ldots + (-1)^{n+1} \text{pl}(A_1 \cup \ldots \cup A_n)$$

per $n = 2$:

$$\text{pl}(A_1 \cap A_2) \leq \text{pl}(A_1) + \text{pl}(A_2) - \text{pl}(A_1 \cup A_2)$$

come dire: la plausibilità che il problema della macchina siano la benzina e al tempo stesso la batteria è la plausibilità di ognuna, ma tolta la plausibilità che il problema sia dovuto a uno o l'altro.

Dalla proprietà segue che:

$$\text{pl}\left(A \cap \overline{A}\right) = \text{pl}(U) = 1 \leq \text{pl}(A) + \text{pl}\left(\overline{A}\right)$$

La completa ignoranza si esprime come

$$\text{bel}(A) = \text{bel}\left(\overline{A}\right) = 0 \qquad\qquad \text{pl}(A) = \text{pl}\left(\overline{A}\right) = 1$$

Basic Mass Assignement Le misure bel e pl esprimono bene l'incertezza ma non sono facilmente utilizzabili senza definire le loro combinazioni. Per fare ciò si ricorre al *Basic Mass Assignement* (*BMA*) che consente di esprimere *qualsiasi* bel e pl:

$$m : \mathcal{P}(U) \to [0, 1] \qquad\qquad \sum_{A \in \mathcal{P}(U)} m(A) = 1 \qquad\qquad m(\oslash) = 0$$

Il BMA ricorda le probabilità, ma in realtà è un concetto diverso e non va confuso. Si nota come *non* sia una misura fuzzy, quindi sono facili da assegnare senza incorrere in infrazioni di assiomi o proprietà. Si nota che $m(A)$ indica il grado di certezza che $x \in A$ ma non ai suoi sottoinsiemi.

Le espressioni per bel e pl diventano:

$$\text{bel}(A) = \sum_{B \subseteq A} m(B) \qquad \text{pl}(A) = \sum_{B \subsetneq A} m(B) = \sum_{B \cap A \neq \oslash} m(B)$$

La completa ignoranza è espressa come:

$$m(U) = 1 \qquad m(A) = 0, \quad \forall A \neq U$$

Cioè:

$$\text{bel}(A) = 0, \quad \forall A \neq U \qquad \text{pl}(A) = 1, \quad \forall A \neq \oslash$$

come dire: all'inizio la credibilità delle cose sconosciute è pari a 0, mentre la plausibilità di cose conosciute è 1.

7.4.4 Misure di Probabilità

Una misura bel è una probabilità se e solo se:

$$\text{bel}(x) = m(x) \quad \text{e} \qquad \text{bel}(x) \neq 0 \quad \text{se } x \text{ non è un singleton}$$

Quindi incertezza e ignoranza coincidono e non nasce mai contraddizione.

7.5 Applicazioni alla robotica mobile

Nella robotica mobile la percezione del mondo avviene attraverso due tipi differenti di sensori:

esterocettivi per misurare distanze dagli ostacoli

Utilizzati nella localization, nel map building, nell'obstacle avoidance.

propriocettivi per misurare lo spazio percorso

Utilizzati nel trajectory tracking, nel Servo control

I sensori possono essere esatti o reali. L'ambiente può essere noto o sconosciuto, statico o dinamico. Nel caso di ambiente dinamico, alcune aree possono modificarsi (ad es. una porta che si apre) e possono occorrere ostacoli inattesi. In questi casi un modulo di Obstacle Avoidance è sempre presente per rendere il sistema di navigazione più robusto.

L'elaborazione fuzzy della percezione consente la costruzione di:

mappe fuzzy globali a partire da sensori ad ultrasuoni, che consentono:

- Pianificazione di un percorso ottimo su mappe fuzzy
- Estrazione della struttura di un ambiente (mappa topology-based)

mappe fuzzy locali (FLM) sono la rappresentazione locale del mondo correntemente attorno al robot. È caratterizzado da un uso limitato di memoria. Consente la fusione sensoriale e la localizzazione all'interno di una mappa topology-based

orizzonte fuzzy che consentono l'obstacle avoidance

7.5.1 Costruzione di mappe

Un robot mobile spesso necessita di una mappa per costruire un percorso dallo start al goal. Per avere una vera autonomia dobbiamo assumere che nessuna conoscenza "a priori" sia disponibile e che il robot debba affidarsi completamente ai suoi sensori.

Nel *sensore ultrasonico* è trasmesso un "wave burst" e l'eco è ricevuta. La distanza dall'ostacolo è calcolata in base al tempo di volo. Non siamo comunque in grado di stabilire la posizione dell'ostacolo. In alcuni casi l'angolo dell'ostacolo potrebbe deviare l'onda e non consentire la ricezione dell'eco. In altri l'eccessiva distanza fa svanire ogni stima. Inoltre bisogna considerare che il sensore è soggetto a false riflessioni.

Approccio con i fuzzy sets Per motivi computazionali l'ambiente è discretizzato in *celle*. L'ambiente coincide con l'insieme bidimensionale di tutte le celle (U). Vogliamo definire:

- l'insieme delle *celle vuote* (E)
- l'insieme delle *celle occupate* (O)

Vogliamo anche definire una *mappa di navigazione* (N) che contenga tutte le informazioni utili a muoversi nell'ambiente. Nel caso ideale ogni cella dovrebbe essere vuota o occupata (vuota = 0 = bianca, occupata = 1 = nera) L'incertezza sensoriale produce *gray-level maps*.

Mettendo in OR (unione) i risultati delle varie misure si ottiene la mappa delle delle celle vuote (E) e la mappa delle celle piene (O). La mappa di navigazione più semplice può quindi essere ottenuta come:

$$N = \overline{O} \cap E$$

Da queste definizioni si ricavano altri tipi di celle:

ambigue piene e occupate: $A = E \cap O$

indeterminate né piene né occupate: $I = \overline{E} \cap \overline{O}$

sicure "molto" vuote meno le occupate e le indeterminate: $S = E^2 \cap \overline{O} \cap \overline{A} \cap \overline{I}$

insicure non sicure: $M = \overline{S}$

Approccio probabilistico Ogni cella ha solo due stati mutuamente esclusivi: $s(C) = E$ oppure $s(C) = O$. Si ha quindi:

$$P[s(C) = E] = 1 - P[s(C) = O]$$

Il sensore è modellato dalla densità di probabilità condizionata $p[d|r]$ che rappresenta la densità di probabilità di una lettura d supposto che l'ostacolo più vicino sia ad una distanza r. Le false riflessioni sono modellate come *outliers*.

La mappa di navigazione è data da $1 - P[s(C) = O]$ ed è ottenuta dalla applicazione sequenziale della regola di Bayes. Deve essere usato Il teorema di Kolmogorov per reperire le probabilità condizionali necessarie, che costituisce un carico computazionale elevato.

Assunzioni del modello:

1. il processo è avviato ipotizzando uno stato di massima entropia:

$$P[s(C) = E] = P[s(C) = O] = 0.5$$

2. ogni cella è assunta indipendente dalle adiacenti

Queste assunzioni possono produrre delle situazioni paradossali, come la presenza di un ostacolo adiacente al robot con probabilità 0.5. Inoltre, la probabilità della esistenza di un ostacolo di una data dimensione dipende dalla dimensione delle celle.

Considerazioni a confronto Comparando i due approcci, osserviamo come, a differenza dell'approccio fuzzy logic, nell'approccio probabilistico sono necessarie più misure per aumentare il valore di $P(O)$. Inoltre basta un solo outlier per riportare $P(O)$ a uno stato di massima entropia ($P(O) = 0.5$)

L'approccio Probabilistico risulta quindi:

- Complesso e "time consuming"
- Troppo sensibile agli outliers
- Lento nella convergenza partendo da completa ignoranza

Nell'approccio Fuzzy sets si ha invece:

- Ampia (troppo?) scelta di operatori
- Una specie di normalizzazione dovrebbe essere introdotta man mano che la conoscenza si accumula
- Buoni risultati
- Implementazione veloce

L'utilizzo delle mappe fuzzy sono la pianificazione del cammino ottimo con l'algoritmo A^*, l'analisi degli ambienti e deduzione di mappe con elementi topologici, la localizzazione. Con la pianificazione del cammino con A^* è possibile trovare un percorso non solo "collision free", ma anche "a rischio minimo" che connetta lo start con il goal.

7.5.2 Mappe "topology-based"

Nelle mappe "topology-based" l'estrazione dell'informazione sulla struttura dello spazio avviene applicando concetti di topologia generale. La descrizione dell'ambiente è in termini di forma dello spazio: spazi aperti connessi da passaggi stretti. L'informazione iniziale sullo spazio libero è rappresentata da una mappa a griglia, che può essere considerata come un grafo dove le stanze sono i nodi, e i collegamenti tra le stanze sono gli archi, con associato un peso per la distanza.

L'algoritmo prevede:

1. Costruzione della mappa a griglia
2. Trasformazione della mappa iniziale per trovare spazi aperti connessi da passaggi stretti (morfologia matematica):
   - Estrae informazioni sulla forma nell'immagine e si basa sui concetti di *Elemento strutturante ES* (definisce la forma da studiare) e di *Filtraggio dell'immagine* con l'ES traslato in tutte le locazioni.
   - Gli operatori fondamentali sono la Dilatazione (allargo le informazioni ottenute) e l'Erosione (il contrario). Tutte le operazioni possono essere definite a partire da un opportuno ES e da una sequenza di operatori fondamentali
3. Segmentazione della mappa in componenti connesse fuzzy (topologia digitale) (individuo le stanze)
4. Costruzione del grafo

7.5.3 Localizzazione

La localizzazione di un luogo (ad esempio un corridoio, un incrocio a T, un angolo, ...) si esegue il seguente procedimento:

1. si collezionano dati usando sensori ad ultrasuoni, laser e visione, mappe fuzzy locali
2. si esegue un primo riconoscimento valutando la similitudine tra i fuzzy set di riferimento e la costruzione del *World mark*[3]
3. se possibile, si esegue il riconoscimento combinato, grazie alle informazioni di altri agenti indipendenti

7.5.4 Obstacle avoidance

Per l'Obstacle avoidance usiamo dei sensori ad ultrasuoni, in particolare 5 "wide beam" range finders con un cono di radiazione di circa 70 gradi, che comportano:

- Grandi spazi con un numero di sensori ridotto
- Elevata incertezza nella collocazione degli ostacoli
- Riflessioni multiple

Si usa la tecnica basata sulla Fuzzy logic conosciuta come "*Metodo degli Orizzonti*". In input si hanno:

- Direzione di navigazione desiderata Θ_{des} e velocità desiderata v_{des}
- Letture sonar
- Informazioni odometriche fornite dagli encoder montati sugli assi dei motori

In output si ha:

- Direzione di navigazione di riferimento Θ_{ref} e velocità di riferimento v_{ref} che garantiscano una navigazione priva di collisioni nella direzione più vicina possibile a quella desiderata

Gli orizzonti fuzzy possono essere usati anche per rappresentare il rischio di prendere una direzione. L'algoritmo inizia con la valutazione, ad ogni passo, dei tre orizzonti:

attrattivo A la compatibilità con la direzione desiderata

repulsivo R il rischio di collisione

[3] il World mark è una rappresentazione fuzzy dell'orizzonte intorno al robot, in cui è estratto, per ogni direzione (quindi 360 gradi), il massimo valore di "occupancy"

cinematico K la compatibilità con i vincoli cinematici del robot

Combinando questi tre si può valutare il costo del movimento in ogni direzione e calcolare un nuovo *orizzonte* D, calcolato come:

$$D(\Theta) = \overline{A}(\Theta) \cup R(\Theta) \cup \overline{K}(\Theta)$$

La direzione di navigazione di riferimento Θ_{ref} è calcolata minimizzando D.

Le letture sensoriali possono cambiare enormemente durante il movimento del robot a causa del largo cono di radiazione. Come conseguenza, il percorso del robot può essere "vagabondante". Ricordare un certo numero di letture passate riduce questo fenomeno, ma rende l'OA più lenta nella reazione. Le vecchie letture sono pesate con la loro età: più vecchie, meno attendibili: le misure più vecchie devono essere rimappate sull'orizzonte attuale tenendo conto anche del movimento del robot.