Smart House

735722 - Mattia Curatitoli
737838 - Danilo Deponti

Introduzione

L'invecchiamento della popolazione ha un impatto sempre più significativo sul sistema sanitario. Per questo si cercano vie più efficienti per portare assistenza in casa delle persone bisognose, in particolare gli anziani, il cui numero secondo le stime é in costante aumento grazie alle migliori condizioni di vita e di cure.
Monitorare le attività umane (`ADL`) è diventato un aspetto fondamentale per costruire un ambiente intelligente atto a garantire il benessere in grado di aumentare sia la sicurezza, sia l'autonomia dei soggetti.
Uno degli approcci più promettenti ed economici è il monitoraggio delle attività tramite una rete wireless di sensori (`WSN`) disposti nell'ambiente abitativo, in quanto molto flessibile e di facile sviluppo. Sono stati sviluppati parecchi modelli che utilizzano sensori per monitorare le attività `ADL`, come ad esempio *Reti Bayesiane* o *Conditional Random Fields*. In particolare, si è notato che ***Hidden Markov Model (HMM)*** è un modello performante in questo dominio applicativo; alcuni articoli approfondiscono il modello proponendo soluzioni di HMM ibridi, noi invece abbiamo implementato una versione classica.

I sensori

Per costruire e testare il modello, sono stati utilizzati dataset presi da due appartamenti. Ogni appartamento fornisce due dataset: il primo con le rilevazioni generate dai sensori, il secondo con le attività rilevate dai sensori. I sensori utilizzati sono stati di vario tipo (magnetici, a pressione, elettrici etc), disposti in più zone della casa in maniera il meno invasiva possibile e hanno registrato e salvato i dati per diversi intervalli di tempo (due settimane circa).
Il dataset delle rilevazioni sensoristiche (figura 1) è una lista di righe corrispondenti all'intervallo di tempo nei quali i sensori sono stati attivi. Ogni riga é composta da:

```
Start time - End time - Location - Type - Place
```

Start time	End time	Location	Type	Place
2011-11-28 02:27:59	2011-11-28 10:18:11	Bed	Pressure	Bedroom
2011-11-28 10:21:24	2011-11-28 10:21:31	Cabinet	Magnetic	Bathroom
2011-11-28 10:21:44	2011-11-28 10:23:31	Basin	PIR	Bathroom
2011-11-28 10:23:02	2011-11-28 10:23:36	Toilet	Flush	Bathroom
2011-11-28 10:25:44	2011-11-28 10:32:06	Shower	PIR	Bathroom
2011-11-28 10:34:23	2011-11-28 10:34:41	Fridge	Magnetic	Kitchen
2011-11-28 10:34:44	2011-11-28 10:37:17	Cupboard	Magnetic	Kitchen
2011-11-28 10:38:00	2011-11-28 10:42:41	Toaster	Electric	Kitchen
2011-11-28 10:38:33	2011-11-28 10:38:40	Fridge	Magnetic	Kitchen
2011-11-28 10:41:29	2011-11-28 10:41:36	Cupboard	Magnetic	Kitchen
2011-11-28 10:41:43	2011-11-28 10:41:59	Cooktop	PIR	Kitchen
2011-11-28 10:41:59	2011-11-28 10:42:55	Microwave	Electric	Kitchen
2011-11-28 10:49:48	2011-11-28 10:51:13	Basin	PIR	Bathroom
2011-11-28 10:51:41	2011-11-28 13:05:07	Seat	Pressure	Living
2011-11-28 13:06:04	2011-11-28 13:06:06	Basin	PIR	Bathroom

Figura 1: Dataset dei sensori

dove la tripla `Location-Type-Place` indica il sensore che ha effettuato la rilevazione. Nel dataset delle attività ogni riga contiene l'intervallo di tempo e il nome dell'attività rilevata. Ogni attività è strettamente legata alla rilevazione effettuata dai sensori, ed é rappresentata (figura 2) su una riga come:

```
Start time - End time - Activity
```

Start time	End time	Activity
2011-11-28 02:27:59	2011-11-28 10:18:11	Sleeping
2011-11-28 10:21:24	2011-11-28 10:23:36	Toileting
2011-11-28 10:25:44	2011-11-28 10:33:00	Showering
2011-11-28 10:34:23	2011-11-28 10:43:00	Breakfast
2011-11-28 10:49:48	2011-11-28 10:51:13	Grooming
2011-11-28 10:51:41	2011-11-28 13:05:07	Spare_Time/TV
2011-11-28 13:06:04	2011-11-28 13:06:31	Toileting
2011-11-28 13:09:31	2011-11-28 13:29:09	Leaving
2011-11-28 13:38:40	2011-11-28 14:21:40	Spare_Time/TV
2011-11-28 14:22:38	2011-11-28 14:27:07	Toileting
2011-11-28 14:27:11	2011-11-28 15:04:00	Lunch
2011-11-28 15:04:59	2011-11-28 15:06:29	Grooming
2011-11-28 15:07:01	2011-11-28 20:20:00	Spare_Time/TV
2011-11-28 20:20:55	2011-11-28 20:20:59	Snack
2011-11-28 20:21:15	2011-11-29 02:06:00	Spare_Time/TV

Figura 2: Dataset delle attività

HMM

Un *HMM (Hidden Markov Model)* è un modello probabilistico definito da variabili osservabili x_t e variabili nascoste y_t, dove t indica il tempo. In questo caso le variabili osservabili sono i sensori, mentre le variabili nascoste sono le attività e dipendono come in figura 3.

In un `HMM` standard le variabili nascoste y_t (y al tempo t) dipendono solo dallo stato delle variabili y_{t-1} (y al tempo $t-1$), mentre le variabili osservabili x_t (x al tempo t) dipendono solo dallo stato delle variabili y_t (y allo stesso istante t).

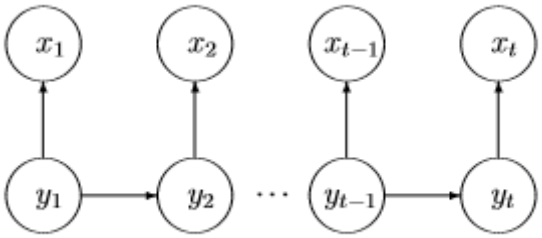

Figura 3: Dipendenze in un Hidden Markov Model

Necessari in un `HMM` sono:

- i dati relativi a $p(y)$, ovvero la probabilità a priori delle variabili y;
- $p(y_t|y_t-1)$ le probabilità di transizione da uno stato y al successivo;
- $p(x_t|y_t)$ le probabilità di una variabile x_t data y_t.

Il Software

Il software è stato sviluppato in `Python`, e permette di creare un `HMM` partendo da una sequenza di rilevazioni prese dal dataset di un appartamento.
Si è deciso di trasformare le rilevazioni dal formato testuale fornito `.txt` in un formato più compatto ed ordinato, il `csv`. Come verrà spiegato dettagliatamente nel paragrafo successivo, il software analizza il file e calcola le probabilità iniziali, di transizione degli stati e di emissione degli eventi in base alle occorrenze all'interno del file. Successivamente, il software si avvarrà di 2 librerie free di nome `GHMM` e `Pomegranate`, in grado di creare modelli `HMM`, fare del *training* su di essi e permette inoltre di inferire sul modello creato.
Per la lettura del file dei sensori, la tripla `Location-Type-Place` è stata utilizzata come chiave univoca per la definizione delle emissioni, che vengono quindi considerate come stati distinti per ogni nuova tripla rilevata all'interno del file.

Le funzioni

Di seguito sono descritte le funzioni principali del software, in modo da illustrare dettagliatamente il metodo usato per creare le probabilità utilizzate per la creazione del modello.

check_and_generate_csv

Prende in input il path di un file testuale `.txt` ed esamina riga per riga (evitando le prime due che nel dataset fornito sono di intesazione) e per ognuna verifica che lo `Start time` sia antecedente al `End time`; sono stati anche implementati, ma non funzionanti, i controlli tra gli `Start time` e `End time` tra righe consecutive. Se non sono presenti errori genera un file `.csv` con i dati controllati, altrimenti lancia un messaggio di errore che indica in quale righe sono state rilevate incorrettezze.

normalize_list e normalize_matrix

Queste funzioni prendono in input rispettivamente liste e matrici e normalizzano i dati contenuti, utilizzando la divisione fornita della libreria `numpy`. In questo modo ogni lista e ogni riga della matrice ha come somma 1.0.

csv_list e csv_matrix

Come facilmente intuibile, generano file in formato `.csv` a partire dai dati in input: lista dei nomi delle variabili (su righe e colonne per la matrice) e lista (o matrice) dei valori da inserire.

obtain_p_adls

Permette di ricavare la lista delle attività rilevate e le probabilità iniziali di ogni singola attività. Questa funzione prende in input il percorso del file csv contenente le attività da analizzare e semplicemente conta le occorrenze di ciascuna stessa attività all'interno del file incrementando un contatore ad ogni rilevazione, dopodichè normalizza i dati.

obtain_t_adls

Permette di ricavare la matrice delle transizioni per le `ADLs`. Conta, ogni volta che un attività accade al tempo t, quante volte ogni singola attività compare al tempo $t-1$, in modo da generare una matrice $n \times n$ con la somma, per ogni attività, delle volte che le altre attività sono accadute successivamente. Infine normalizza i dati.

obtain_list_sens

A partire dal file in input ricava la lista dei sensori attivati durante le rilevazioni (composti dalla tripla univoca `Location-Type-Place`) e l'elenco con cui sono stati attivati.

obtain_o_sens_adls

La funzione permette di ricavare le probabilità di emissione dalle `ADLs` ali sensori. Viene tenuto conto di tutte le volte che l'intervallo di tempo di una rilevazione del sensore

cade all'interno della rilevazione di un'attività. La matrice ottenuta dovrà infine essere normalizzata.

Librerie utilizzate

Come precedentemente detto, è stato deciso di utilizzare l'approccio con diverse librerie. Con `GHMM` è stato creato il modello `HMM` ed è stata fatta inferenza utilizzando `Viterbi`, ma le analisi più dettagliate sui risultati sono state fatte con `Pomegranate`.

GHMM

La libreria `GHMM`, dati un alfabeto di emissioni `sigma`, una distribuzione discreta su di esso, la matrice di transizione delle `ADLs`, le loro probabilità iniziali e le probabilità di emissione dei sensori per ogni attività, permette di creare un `HMM`. Successivamente, con `Viterbi`, data una sequenza di rilevazioni di sensori viene calcolato quanto il modello è abile a rilevare l'attività corretta.

Pomegranate

`Pomegranate` è un package che permette di creare modelli probabilistici dalle `Reti Bayesiane` a `HMM`. E' un'estensione del package `YAHMM`. Per utilizzare un `HMM` è necessario chiamare la funzione:

```
model = HiddenMarkovModel( name="Model-name" )
```

Il package caratterizza ogni stato nascosto e le sue emissioni con la seguente sintassi:

```
ADL1 = State( DiscreteDistribution( 'sens1':  0.1, 'sens2':  0.4,
'sens3':  0.5 ), name='ADL1' )
```

Dopodichè, è necessario definire le probabilità iniziali come segue:

```
model.add_transition( model.start, ADL1, 0.6 )
```

e le probabilità di transizione tra gli stati nel seguente modo:

```
model.add_transition( ADL1, ADL2, 0.65 )
```

Successivamente, dopo la chiamata `model.bake()`, che permette l'effettiva elaborazione dei dati forniti per la creazione del `HMM` è possibile fare diversi tipi di inferenze sul modello. Come `forward`,`backward`,`forward-backward` e `Viterbi`.

Esecuzione e stime

Una volta generato il nostro modello, utilizzeremo `Viterbi` dando in input al comando `model.viterbi(sequence)` la sequenza di sensori utilizzata in origine per stimare le probabilità del modello. In questo modo possiamo verificare se la sequenza di `ADLs` risultante è coerente con quella della `Grounf Truth`.

Parametri di affidabilità

(da finire una volta implementati i metodi)