Capitolo 1

Introduzione

1.1 Inquadramento generale

L'analisi e la comprensione di grosse quantità di dati in questi ultimi anni è diventata una pratica fondamentale per comprendere i fenomeni che ci circondano. In particolare quando questi fenomeni sono descritti da una posizione geografica e da un momento temporale preciso ci permettono di realizzare delle deduzioni che altrimenti sarebbero impensabili.

In questo ramo della ricerca si inserisce il *Data Mining*, ovvero l'insieme di tecniche e metodologie che hanno per oggetto l'estrazione di informazioni utili da grandi quantità di dati attraverso metodi automatici o semi-automatici (*machine learning*).
Nel lavoro svolto in particolare viene effettuata una *analisi delle associazioni*, una tecnica particolare di data mining utilizzata per la ricerca di connessioni di eventi con determinate caratteristiche.

1.2 Breve descrizione del lavoro

L'algoritmo preso in esame è lo *Spatio-Temporal Breath-first Miner (STB-FM)* definito da Piotr S. Maciag and Robert Bembenik. Questo particolare algoritmo permette di definire delle sequenze di tipologie di eventi connesse nello spazio e nel tempo. Viene definito un vicinato basato su un raggio spaziale e un intervallo di tempo dal quale valutare se il singolo evento è "vicino" ad altri. Questa valutazione viene fatta per tutti gli eventi di uno stesso tipo

rispetto a tutti gli eventi di un altro tipo. Da queste valutazioni si ricava un valore di *connessione* tra tipi compreso tra $[0, 1]$, più questo valore tende a 1 più i rispettivi tipi sono associati.
Questo lavoro viene fatto su diverse combinazioni di tipi, queste combinazioni vengono unite in sequenze e a ogni sequenza viene associato il valore di associazione.
es.

$[A, B, C]$ - 0.8
$[B, D]$ - 0.5

Tramite questi valori dovremmo essere in grado di capire quanto degli eventi sono associati e, in questi casi, cercare di prevenire/supportare (a seconda dei casi) l'evento di tipologia successiva.

Il caso preso in esame è quello dei crimini avvenuti a Boston, utilizzando il database fornito dal Boston Police Department (BPD) in cui sono registrati tutti i crimini avvenuti a Boston dal 2015, etichettandoli con la tipologia (di crimine), le coordinate GPS dell'evento e il momento in cui avvengono, con il giorno e l'orario.

es.

Offence Code	Date	Lat	Long
LarcFromMotVehic	01/01/2018 00:00	4.235.314.550	-7.107.763.936
ResidentialBurglary	01/01/2018 00:00	4.229.755.533	-7.105.970.910
AggravatedAssault	01/01/2018 02:23	4.235.040.583	-7.106.512.526

Tabella 1.1: esempio eventi

Esso rispetta tutti i vincoli di applicazione di questo algoritmo, vi è un gran numero di eventi etichettati per tipologia, geolocalizzati spazialmente e temporalmente, pertanto è stato scelto per l'applicazione pratica.

1.3 Scopo e prospettive

Lo scopo di questo lavoro è quindi quello di implementare il l'algoritmo *Spatio-Temporal Breath-First Miner (STBFS)* per capire le sue applicazioni

a casi concreti come quello dei crimini di Boston e analizzarne l'efficacia anche in termini di tempi di computazione.
Esso si apre a possibili sviluppi futuri anche in contesti completamente diversi rispetto a quello preso in esame, come ad esempio l'analisi dell'incidenza di epidemie.

1.4 Struttura della tesi

La tesi è strutturata nel seguente modo:

- Nel **capitolo due** si parla della base teorica su cui si basa l'algoritmo, in particolare i calcoli che si effettuano e la struttura dati utilizzata nel paper (anche possibili alternative come algoritmo apriori?)
- Nel **capitolo tre** si analizza in modo più approfondito il dataset utilizzato e le varie considerazioni fatte
- Nel **capitolo quattro** si parla dell'implementazione effettuata
- Nel **capitolo cinque** si analizzano i risultati ottenuti sia in termini di tempi di computazione che in termini di significato degli stessi
- **Conclusioni** e prospettive future

Capitolo 2

Stato dell'Arte

2.1 Paper

Come precedentemente anticipato l'algoritmo oggetto di questo lavoro è
A Novel Breadth-first Strategy Algorithm for Discovering Sequential Patterns from Spatio-temporal Data
di Piotr S. Maciag e Robert Bembenik del *Instituite of Computer Science, Warsaw University of Tecnology, Nowowiejska 15/19, 00-665, Warsaw, Poland.*
Di seguito vi è la trascritta la base teorica su cui si fonda l'algoritmo e le strutture dati utilizzate per la sua realizzazione.

2.2 Vicinato

Il problema che si considera è quello della scoperta di pattern da un certo dataset di istanze di eventi, i quali sono di una certa tipologia, definiamo quindi:

$D \rightarrow$ dataset di istanze di eventi
$F \rightarrow$ insieme di tipologie di eventi

Ogni istanza $e \in D$ ha:

- chiave di identificazione (unica)
- location spaziale (es. coordinate geografiche)
- istante temporale

- tipologia $f \in F$

La sequenza di eventi (pattern) è così definita:

$\overrightarrow{s} = f_{i_1} \rightarrow f_{i_2} \rightarrow ... \rightarrow f_{i_n}$, dove $f_{i_1}, f_{i_2}, ..., f_{i_n} \in F$

Quindi per ogni due tipologie di eventi consecutive in una sequenza $f_{i_{j-1}} \rightarrow f_{i_j}$, le istanze dell'evento $j-1$ sono connesse con con il tipo successivo spazialmente e temporalmente.

L'insieme di eventi collegati in questo modo a una determinata istanza viene definito **neighborhood** o vicinato.

Esempio

Consideriamo una situazione come quella in Fig. 2.1, dove:

$D = \{a1, a2, b1, b2, b3, b4, b5, b6, b7, b8, c1, c2, c3, c4\}$

$F = \{A, B, C\}$

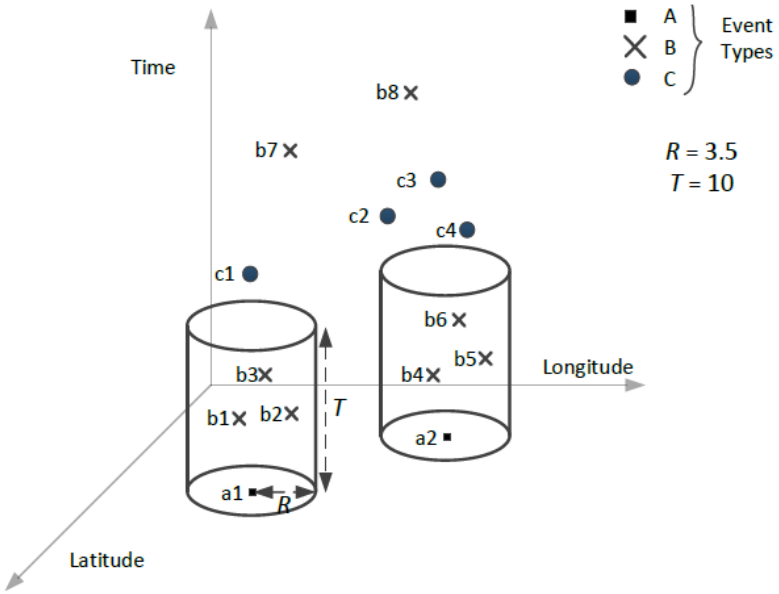

Figura 2.1: esempio di istanze con vicinato degli eventi di tipo A

Una sequenza significativa per esempio potrebbe essere $\overrightarrow{s} = A \rightarrow B \rightarrow C$. Per valutare la connessione $A \rightarrow B$ bisogna considerare il *neighborhood* tra le loro istanze. Come si nota dalla figura è stato scelto un raggio spaziale pari a $R = 3.5$ e un intervallo temporale pari a $T = 10$ per la dimensione del vicinato.

2.3 Nozioni di base

Dopo aver inquadrato graficamente il problema ora definiamo formalmente i concetti base usati per il calcolo di tutte le sequenze pattern usati nell'algoritmo.

Spazio Neighborhood Con $V_{N(e)}$ denotiamo lo spazio di neighborhood (vicinato) dell'istanza e. Questo spazio si basa su tre dimensioni, che sono le due dimensioni spaziali - latitudine e longitudine - e la dimensione temporale. Graficamente ne risulta un cilindro, con parametri R che denota il raggio spaziale e T l'intervallo temporale.

In Figura 2.2 vengono mostrati i due neighborhood tratti dall'*esempio* della Figura 2.1: $V_{N(a1)}$ e $V_{N(a2)}$.

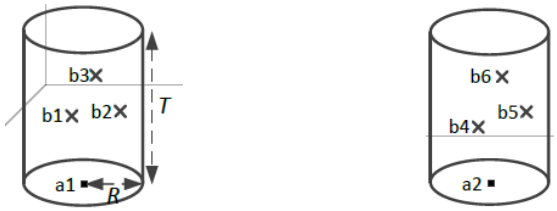

Figura 2.2: $V_{N(a1)}$ *e* $V_{N(a2)}$

Neighborhood rispetto a una tipologia di evento Data una certa istanza e, il *neighborhood* di e è definito nel modo seguente:

$$\begin{aligned} N_f(e) = \{e | p \in D(f) \\ \wedge\ distance(p.location, e.location) \leq R \\ \wedge\ (p.time - e.time) \in [0, T]\} \end{aligned}$$

dove R e T sono i parametri dello spazio di vicinato $V_{N(e)}$ e $D(f)$ è l'insieme di istanze degli eventi di tipo f nel dataset D.
Nota si considerano solo gli eventi che si susseguono dal punto di vista temporale, in quanto è poco significativo considerare gli eventi passati dell'istanza nella ricerca di sequenze.

Riassumento con $N_f(e)$ denoto l'insieme di istanze di tipo f contenute all'interno dello spazio $V_{N(e)}$.

Nel nostro *esempio* della Figura 2.2:
$N_B(a1) = \{b1, b2, b3\}$ e $N_B(a2) = \{b4, b5, b6\}$

Set di istanze Per una sequenza di tipi di eventi $\overrightarrow{s} = \overrightarrow{s}[1] \rightarrow \overrightarrow{s}[2] \rightarrow ... \rightarrow \overrightarrow{s}[m]$ di lunghezza m, gli insiemi (set) di istanze $I(\overrightarrow{s}[1]), I(\overrightarrow{s}[2]), ..., I(\overrightarrow{s}[m])$ che sono inclusi nella sequenza $\overrightarrow{s}$ sono definiti come segue:

1. Per un tipo di evento $\overrightarrow{s}[1]$, il set di istanze $I(\overrightarrow{s}[1])$ è definito come:
$$I(\overrightarrow{s}[1]) = D(\overrightarrow{s}[1])$$
2. Per i tipi $\overrightarrow{s}[2] \rightarrow ... \rightarrow \overrightarrow{s}[m]$ con $i = 2, 3, ..., m$, gli insiemi di istanze $I(\overrightarrow{s}[i]$ sono definiti così:
$$I(\overrightarrow{s}[i]) = distinct(\bigcup_{e \in I(\overrightarrow{s}[i-1])} N_{\overrightarrow{s}[i]}(e))$$

In pratica per il primo tipo di evento (d'ora in poi nominato solo "tipo") che partecipa alla sequenza $\overrightarrow{s}$, il set di istanze $I(\overrightarrow{s}[1])$ corrisponde al set di istanze di tipo $\overrightarrow{s}[1]$ in D, ovvero $D(\overrightarrow{s}[1])$.
Per i tipi successivi di $\overrightarrow{s}$, i set $I(\overrightarrow{s}[i])$ sono definiti come insiemi di istanze contenute nei vicinati di istanze a partire da $I(\overrightarrow{s}[i-1])$.
Seguendo questo meccanismo si valuta tutta la sequenza e tendendo in considerazione l'insieme di istanze calcolato al passaggio precedente.

Consideriamo la sequenza $\overrightarrow{s} = A \rightarrow B$ dal dataset dell'*esempio* in Figura 2.1. In questo caso avremmo i seguenti set di istanze:
$I(\overrightarrow{s}[1]) = \{a1, a2\}$
$I(\overrightarrow{s}[2]) = \{b1, b2, b3, b4, b5, b6\}$

Participation Ratio Data una sequenza $\overrightarrow{s} = \overrightarrow{s}[1] \rightarrow \overrightarrow{s}[2] \rightarrow ... \rightarrow \overrightarrow{s}[m]$ il *participation rateo* tra due tipi consecutivi contenuti in $\overrightarrow{s}$ è definito:

$$PR(\overrightarrow{s}[i-1] \rightarrow \overrightarrow{s}[i]) = \frac{|I(\overrightarrow{s}[i])|}{|D(\overrightarrow{s}[i])|}$$

questo valore corrisponde al numero di istanze distinte di tipo $\overrightarrow{s}[i]$ contenute nei neighborhoods delle istanze di tipo $\overrightarrow{s}[i-1]$ diviso il numero di istanze di tipo $\overrightarrow{s}[i]$ presenti nel dataset D.
Per ogni coppia di tipi consecutivi $(\overrightarrow{s}[i-1], \overrightarrow{s}[i])$ in una sequenza $\overrightarrow{s}$ il *participation rateo* è definito come il rapporto tra $|I(\overrightarrow{s}[i])|$ e $|D(\overrightarrow{s}[i])|$ e il suo valore è compreso nel range $[0, 1]$.

Participation Index Data una sequenza lunga m: $\overrightarrow{s} = \overrightarrow{s}[1] \rightarrow \overrightarrow{s}[2] \rightarrow ... \rightarrow \overrightarrow{s}[m]$, il *participation index* è così definito:

1. se $m = 2$:
$$PI(\overrightarrow{s}) = PR(\overrightarrow{s}[1] \rightarrow \overrightarrow{s}[2])$$
2. se $m > 2$:
$$PI(\overrightarrow{s}) = min \begin{cases} PI(\overrightarrow{s}^*) \\ PR(\overrightarrow{s}[m-1] \rightarrow \overrightarrow{s}[m] \end{cases}$$

dove $\overrightarrow{s}^* = \overrightarrow{s}[1] \rightarrow \overrightarrow{s}[2] \rightarrow ... \rightarrow \overrightarrow{s}[m-1]$

Il *participation index* corrisponde al minimo di tutti i *participation rateo* calcolati su due tipi consecutivi presenti in $\overrightarrow{s}$, ed è il nostro valore di output dell'algoritmo.
Intuitivamente lo possiamo pensare come il numero di istanze di eventi collegate alle istanze di tipi connessi precedentemente secondo un ordine stabilito dalla sequenza e questo ci dà la misura di quanto la sequenza sia correlata.

Consideriamo il nostro *esempio* della Figura 2.1, per la sequenza $\overrightarrow{s} = A \rightarrow B$ il $PI(\overrightarrow{s}) = 0.75$ ($PI(A \rightarrow B) = PR(A \rightarrow B) = \frac{6}{8} = 0.75$) che è un buon risultato di correlazione.

Capitolo 3

Algoritmo

3.1 STM Algorithm

Definiti i concetti teorici di base ora verrà descritto l'algoritmo Spatio-Temporal Miner che permette di trovare tutte le sequenze pattern con *participation index* maggiori di una certa soglia **threshold** θ data dall'utente.
La definizione di un θ permette di considerare come risultato solo le sequenze più significative e ignorare dalla computazione, invece, quelle che hanno un livello di correlazione troppo basso.
Naturalmente bisogna considerare che l'utente definisce il parametro θ ma anche il raggio spaziale R e l'intervallo temporale T.

3.2 Albero - SPTree

A sostegno dell'algoritmo viene definita una struttura ad albero dedicata per ridurre drasticamente il numero delle computazioni, in particolare per quanto riguarda la generazione di sequenze candidato nuove.

INSERIRE DEFINIZIONE DI ALBERO DAL CORMAN

Consideriamo un insieme di tipi di eventi di questo tipo: $F = \{A, B, C, D, E, F\}$ e un insieme L di sequenze pattern presentate in Tabella 2.1.

Algorithm 1: Spatio-temporal breath first miner for discovering significant sequential patterns (STBFM)

Data: D - dataset delle istanze degli eventi, F - insieme di tipi di eventi, R - raggio spaziale, T - intervallo temporale, θ - threshold

Result: *Top* - insieme delle N sequenze più significative

```
1 C_2 := generazione candidati di lunghezza 2
2 L_2 := VerifyCAndidates(C_2)
3 k = 3
4 while L_{k-1} ≠ θ do
5     C_k := CandidateGen(C_2)
6     L_k := VerifyCandidates(C_k)
7     Add L_k to L
8     k := k + 1
9 return L
```

Algorithm 2: VerifyCandidates

Data: C_m - insieme di candidati sequenze di tipi di lunghezza m, θ - threshold

Result: L_m - insieme di sequenze di tipi significativi di lunghezza m

```
1 C_m := ∅
2 foreach s⃗ ∈ C_m do
3     CalculatePI(s⃗)
4     if PI(s⃗) ≥ θ then
5         Add s⃗ to L_m
6 return L_m
```

Algorithm 3: CandidateGen(L_{m-1})

Data: L_{m-1} - insieme di sequenze di tipi di lunghezza $m-1$
Result: C_m - insieme di candidati sequenze di tipi di lunghezza m

$C_m := \emptyset$
foreach $\overrightarrow{s_i} \in L_{m-1}$ **do**
  foreach $\overrightarrow{s_j} \in L_{m-1} \wedge \overrightarrow{s_i} \neq \overrightarrow{s_j}$ **do**
    if $\overrightarrow{s_i}[2] = \overrightarrow{s_j}[1] \wedge \overrightarrow{s_i}[3] = \overrightarrow{s_j}[2] \wedge ... \wedge \overrightarrow{s_i}[m] = \overrightarrow{s_j}[m-1]$ **then**
      $\overrightarrow{s} := \overrightarrow{s_i}[1] \rightarrow \overrightarrow{s_i}[2] \rightarrow ... \rightarrow \overrightarrow{s_i}[m-1] \rightarrow \overrightarrow{s_j}[m-1]$
      $I(\overrightarrow{s}[1]) := I(\overrightarrow{s_i}[1]) \wedge I(\overrightarrow{s}[2]) := I(\overrightarrow{s_i}[2]) \wedge ... \wedge$
      $I(\overrightarrow{s}[m-1]) := I(\overrightarrow{s_i}[m-1]) \wedge I(\overrightarrow{s}[m]) :=$
       CalculateNeighborhood$(I(\overrightarrow{s_i}[m-1]), I(\overrightarrow{s_j}[m-1])$
      Add $\overrightarrow{s}$ to C_m
return C_m

Algorithm 4: CalculatePI($\vec{s}$)

Data: $\overrightarrow{s} = \overrightarrow{s}[1] \rightarrow \overrightarrow{s}[2] \rightarrow ... \rightarrow \overrightarrow{s}[m-1] \rightarrow \overrightarrow{s}[m]$ - una sequenza di tipi di eventi

return
$min(PI(\overrightarrow{s}[1] \rightarrow \overrightarrow{s}[2] \rightarrow ... \rightarrow \overrightarrow{s}[m-1]), PR(\overrightarrow{s}[m-1] \rightarrow \overrightarrow{s}[m])$

Algorithm 5: CalculateNeighborhood($I(\vec{s_i}[m-1]), I(\vec{s}[m-1])$)

Data: $I(\overrightarrow{s_i}[m-1])$ - un insieme di istanze di tipo $\overrightarrow{s}[m]$ della sequenza $\overrightarrow{s_i}$, $I(\overrightarrow{s_j}[m-1])$ - un insieme di istanze di eventi di tipo $\overrightarrow{s_j}[m]$ della sequenza $\overrightarrow{s}_j$
Result: $I(\overrightarrow{s}[m])$ - un insieme di istanze di eventi di tipo $\overrightarrow{s}[m]$ della sequenza candidato $\overrightarrow{s}$

return $I(\overrightarrow{s}[i])$ as $distinct(\bigcup_{e \in I(\overrightarrow{s}[i-1])} N_{\overrightarrow{s}[i]}(e))$

F	A, B, C, D, E, F
L	Sequenze pattern
L_2	$A \to B$, $B \to C$, $B \to D$, $C \to E$, $C \to F$.
L_3	$A \to B \to C$, $A \to B \to D$, $B \to C \to E$, $B \to C \to F$.
L_4	$A \to B \to C \to E$, $A \to B \to C \to F$.

Tabella 3.1: esempio di sequenze pattern

Partendo dalla radice verranno inserite tutte le sequenze significative secondo i criteri che seguono.

La **root** avrà come figli tutti i tipi di eventi presenti in F. Assumiamo che le sequenze di lunghezza 2 (L_2) siano state generate dai tipi di partenza effettuando tutte le combinazioni tra di essi, con le verifiche del participation rateo come presentato dal L_2 dell'albero in Figura 2.3.

Per ogni sequenza, quindi ogni nodo dell'albero, utilizziamo tre strutture di dati: *firstParent*, *secondParent* e *children*.
Per la generazione dei **nodi** si procede per livello, ovvero ci si basa sul livello immediatamente precedente e si genera quello sottostante, seguendo queste procedure:

- se la sequenza $\overrightarrow{s}$ è stata creata unendo i tipi di eventi f_{i_1} e f_{i_2} a $\overrightarrow{s} = f_{i_1} \to f_{i_2}$, allora:
  $firstParent(\overrightarrow{s}) := f_{i_1}$
  $secondParent(\overrightarrow{s}) := f_{i_2}$
  $\overrightarrow{s}$ viene aggiunta ai $children(\overrightarrow{s})$
- se la sequenza $\overrightarrow{s}$ è stata creata dalle sequenze $\overrightarrow{s}_i$ e $\overrightarrow{s}_j$, allora:
  $firstParent(\overrightarrow{s}) := \overrightarrow{s}_i$
  $secondParent(\overrightarrow{s}) := \overrightarrow{s}_j$
  $\overrightarrow{s}$ viene aggiunta ai $children(\overrightarrow{s})$

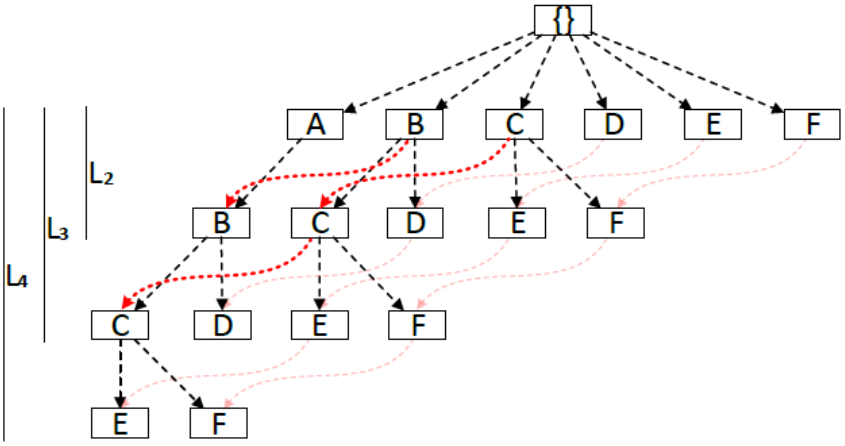

Figura 3.1: SPTree basato sulle sequenze della Tabella 2.1

Esempio Consideriamo la sequenza $A \rightarrow B$ dalla Tabella 2.1 e quindi la sua memorizzazione nel *SPTree* come in Figura 2.3. Assumiamo che dobbiamo generare i candidati di lunghezza 3 quindi generare il terzo livello dell'albero.
In questo caso $A \rightarrow B$ può essere estesa con due diversi tipi C o D, in quanto $A \rightarrow B$ ha come *secondParent* il tipo B ed esso ha come sequenze figlio (*children*) $B \rightarrow C$ e $B \rightarrow D$, pertanto le posso aggiungere alla sequenza $A \rightarrow B$. Seguendo questo processo viene generato l'*SPTree*.

3.3 STBFM

dfghj

3.4 Alternativa - Algoritmo apriori