Inferenza con Junction Trees nei Belief Networks

Francesco Baiocchi
Graduating at University of Florence
AI Homework Assignment.

Abstract

La relazione tratta la tematica dell'inferenza attraverso Junction Trees nelle belief networks, considerando queste nella loro versione di grafi orientati data da J.Pearl.
In un primo momento si presenta il problema dell'inferenza probabilistica e le sue finalità, proseguendo poi per chiarire quale algoritmo di inferenza verrà presentato.
Successivamente la relazione prosegue confrontando i risultati trovati attraverso l'utilizzo di quest'ultimo per lo scambio di messaggi nella rete e poi con quelli attraverso l'uso della definizione di probabilità condizionale.

"PROBABILITY DOES NOT EXIST", meaning that probability does not exist in an objective sense. Rather, probability exists only subjectively within the minds of individuals."
De Finetti, B. (1974), Theory of Probability, Vol. 1. New York: John Wiley and Sons

Le "beliefs" sono un'interpretazione del concetto di probabilità, possono essere viste come un prolungamento di logica di proposizione che permette ragionare con ipotesi ovvero con le proposizioni la cui verità o falsità sono incerte.
Il procedimento di inferenza su una Belief Network punta a calcolare la probabilità di una Query data una certa evidenza ,**P(Query| Evidence)** dove per evidenza si intende la conoscenza dello stato di una o più variabili.
L'inferenza probabilistica permette quindi di utilizzare le informazioni che si hanno a disposizione per calcolare le modifiche sulla probabilità congiunta e quindi sui singoli belief delle variabili. Putroppo sebbene questo problema sia facile da spiegare,è al contrario abbastanza difficile da calcolare.
Questo è infatti un problema **NP-HARD**.
Il particolare algoritmo che viene presentato nella relazione è quello che fa utilizzo della struttura dei Junction Tree.

Introduction

Partendo da una data Belief Network, la costruzione di un Junction Tree per il problema dell'inferenza sulle reti Bayesiane non è diversa dalla costruzione di questo in altri ambiti di applicazione , come ad esempio i Constraint Problems.
Le uniche differenze nel processo di costruzione di questo sono volte a rispettare le proprietà che i Junction Trees devono rispettare affinchè l'inferenza probabilistica funzioni.

- I nodi sono clusters di variabili.
- L'unione di tutti i suoi nodi deve dare l'universo delle variabili del rete iniziale.
- I nodi e i separatori contengono le tabelle di probabilità.
- Venga rispettata la proprietà di Running Intersection.
- Per ogni variabile in U ci dovrà essere un cluster che la contiene insieme ai suoi padri. Non è possibile rompere le famiglie.

Viene aggiunta infatti una fase di "Moralizzazione" della Belief Network, che consente il rispetto dell'ultima prioprietà, fondamentale per la definizione delle Belief Network stessa.
. (Dettagli sull'algoritmo di costruzione del Junction Tree per inferenza probabilistica : [2] Jensen 1997)

La proprietà fondamentale della "Running Intersection" è garantita dal fatto che venga usato un MST(Max Spanning Tree) , mentre il processo di Moralizzazione,connessione di **tutti** i padri con tutti , garantisce la proprietà della famiglia.
Una volta costruito un Junction Tree che rispetti le proprietà elencate , il problema dell'inferenza probabilistica diventa il problema dello scambio di messaggi tra i vari nodi del Junction Tree.
Questo scambio vuole essere bidirezionale e avviene tramite le tabelle di proprietà associate ai cluster e ai separatori, chiamate **CPT = Conditional Probability Table** Dietro queste c'è infatti un'algebra, molto vicina all'algebra relazionale che consente la propagazione di messaggi.

Tutto ciò che serve per implementare lo scambio dei messaggi sono le due operazioni di Prodotto tra tabelle e la Marginalizzazione.

Algoritmo di Propagazione dei Beliefs

Lo scambio dei messaggi in un Junction Tree per le Belief Networks avviene attraverso una procedura chiamata **Absorption**. Questa procedura descrive il passaggio di informazione tra due cluster del Junction Tree.
Il nodo separatore ,che rappresenta l'intersezione dei due cluster,viene informato dal nodo che porta il messaggio.Quest'ultimo infatti si "marginalizza" sulle variabili del separatore,portando cosi l'informazione che il separatore sa gestire. Il separatore stesso poi sarà in grado di trasmetterla all'altro cluster dell'arco. Guardandolo quindi da un contesto più ampio, questo **"messaggio"** di informazione consente la possibilità di propagare i beliefs in tutta la rete , ripetendo la procedura di Absorption più volte.

L'idea, dopo aver capito come scambiare un messaggio tra due cluster, è quella di fissare un radice per il Junction Tree. Questa corrisponde al cluster su cui si vuole fare evidenza. Fissando la radice di conseguenza si stabilisce un ordinamento sugli archi. A questo punto si punto fa transitare l'informazione dalla radice fino alle foglie del Junction Tree (**Distribute Evidence**) che risponderanno con un processo inverso di ricollezione dei belief fino alla radice.(**Collect Evidence**).

Implementazione dell'algoritmo in Python

Nella seguente repo Github si trova l'implementazione in Python di questo algoritmo.
`https://github.com/francescobaio/Elaborato_AI`

Questa implementazione viene usata su due esempi molto semplici di Reti Bayesiane di cui viene fornito il link nella repo Github. Il modulo software permette in Python di memorizzare una rete Bayesiana fornita e il suo rispettivo Junction Tree.

Inferenza probabilistica

Vediamo quindi come ,partendo da una Rete Bayesiana e poi passando al suo rispettivo Junction Tree (entrambi allegati) ,procede l'inferenza probabilistica.

Definizione Joint Probability vs Algoritmo di Inferenza Junction Tree

Come presentato all'inizio della relazione l'obiettivo è quello di calcolare la probabilità di una Query data un 'evidenza. La distribuzione congiunta della rete Bayesiana deriva , dalla sua definizione , dalla produttoria delle CPT $\mathbf{P(U) = \prod_{i=1}^{n} P(X(i) | Pa(X(i))}$. Calcolare la distribuzione congiunta quindi significa eseguire una produttoria di tabelle,dove la tabella risultante sarà definita sull'universo delle variabili U. Una volta calcolata la probabilità congiunta è facile capire che sottoponendo un'evidenza (e) , ovvero la conoscenza dello stato di una o più variabili, si può usare la marginalizzazione di questa sulle variabili rimanenti (U \Q ∪ e) per calcolare le probabilità condizionate.

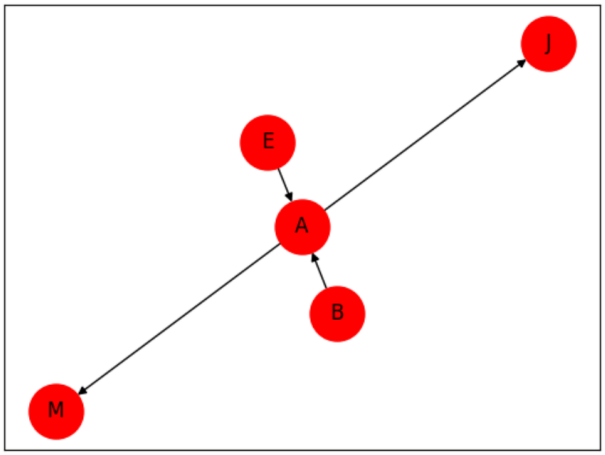

Figure 1: Belief Network del file earthquake

Con il modulo software graphic.py è possibile graficare la Belief Network ottenendo questo risultato,dove i nodi hanno come nome le iniziali dei veri nomi. Si può cosi anche graficare il Junction Tree associato utilizzando il metodo **junction_tree(G)** della libreria **"networkx"**

Tuttavia però, questo è molto costoso nell'implementazione.Questo metodo è supportabile solo in reti piccoli dove la matrice della probabilità congiunta continua ad avere dimensioni gestibili.

Per questo motivo viene usato maggiormente l'algoritmo del Junction Tree.
L'implementazione di questo algoritmo viene fatta utilizzando per ogni Cluster e Separatore una CPT, una matrice con le possibili interpretazioni e le probabilità associate a ognuno di queste. L'inizializzazione viene eseguita dalla funzione **" initialitation() "**.

Attraverso la proprietà di consistenza globale,data dall'Absorption, l'idea del programma è quella di fare evidenza decidendo una radice per il Junction Tree,utilizzando quindi le funzioni **distribute_evidence()** e **collect_evidence()**, per collezionare e distrubuire belief consistenti in tutta la rete rendendola informata completamente dell'evidenza.
L'algoritmo per usare il codice sarebbe quindi :

Algorithm 1 Inference

```
1: scelta di una root,cluster di cui abbiamo evidenza
2: find_leafs(self,root)
3: collect_evidence(self,root)
4: distribute_evidence(self,root)
```

Risultati Sperimentali

I risultati sperimentali sulle due reti scelte non vengono presentati per mancanza di dettagli implementativi. Tuttavia l'intuizione è che per come sono state scelte le reti d'esempio è quella di non riuscire a percepire minimamente la differenza di utilizzo tra i due algoritmi. Allo stesso tempo

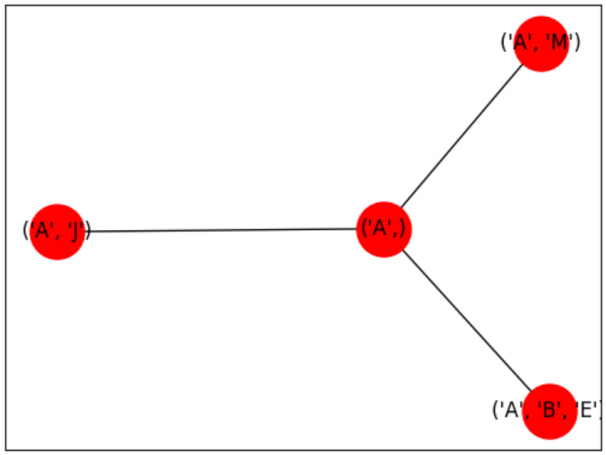

Figure 2: Junction Tree associato alla Belief Network

però se la dimensione della rete crescesse , scegliendo una tra `https://raw.githubusercontent.com/ncullen93/pyBN/master/data/munin.bif` o `https://github.com/ncullen93/pyBN/blob/master/data/andes.bif` si intuisce che la matrice di Joint Probability aumenta notevolmente di volume.
Al contrario invece,utilizzando l'algoritmo di inferenza con Junction Trees, aumentano il numero di CPT di Separatori e Cluster ma le loro dimensioni rimangono sempre contenute.La procedura di scambio di messaggi tiene inoltre informata sempre la rete.

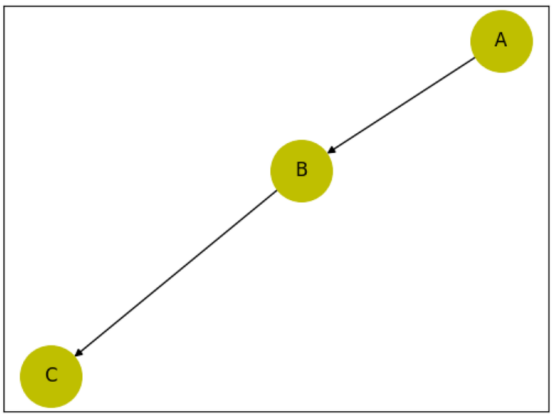

Figure 3: Belief network associato al file simple .bn

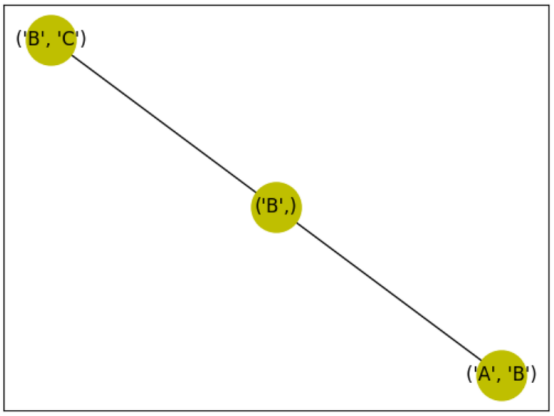

Figure 4: Junction Tree associato alla Belief Network