

Redes bayesianas

Walter Perez Urcia
and

Denis Deratani Mauá

Universidade de São Paulo, Brasil
wperez@ime.usp.br, denis.maua@usp.br

Abstract. Este artigo propõe uma versão do algoritmo de aprendizado de redes bayesianas que usa o score BIC e uma busca no espaço de ordenamentos. O algoritmo tem sido modificado no passo de inicialização de uma permutação dos atributos. Um conjunto de experimentos mostra que fazendo esta modificação no algoritmo obtemos redes melhores para os conjuntos de dados.

A popular and effective method for learning Bayesian network structures is to perform a greedy search on the space of variable orderings followed by an exhaustive search over the restricted space of compatible parent sets. Usually, the greedy search is initialized with a randomly sampled order. In this work we develop heuristics for producing informed initial solutions to order-based search based on the feedback arc set problem.

Categories and Subject Descriptors: I.2.6 [Artificial Intelligence]: Learning

Keywords: redes bayesianas, aprendizado

1. INTRODUÇÃO

2. APRENDIZADO DE REDES BAYESIANAS

O Algoritmo 1 mostra um pseudocódigo da versão original do método de aprendizado de redes bayesianas.

Algoritmo 1: Busca local no espaço de ordenamentos

```
1  L = Permutation( x[ 0 ] , x[ 1 ] , ... , x[ n ] )
2  while k < NUM_RESTARTS :
3      best = find_order( L )
4      if score( best ) > score( L ) :
5          L = best
```

Além disso, a função *find_order* é mostrada a continuação. O método *swap* troca os valores entre as posições i e $i + 1$ de L .

Algoritmo 2: Find_Order

```
1  best_i = 0
2  best_diff = -INF
3  score_orig = score( L )
4  for i in range( n - 1 ) :
5      swap( L , i )
```

Copyright©2012 Permission to copy without fee all or part of the material printed in KDMiLe is granted provided that the copies are not made or distributed for commercial advantage, and that notice is given that copying is by permission of the Sociedade Brasileira de Computação.

```

6         d_i = score( L ) - score_orig
7         if d_i > best_diff :
8             best_diff = d_i
9             best_i = i
10        swap( L , i )
11    swap( L , best_i )
12    return L

```

Na seção 4 este algoritmo será usado para encontrar redes bayesianas para vários conjuntos de dados.

3. MELHORA NO APRENDIZADO DE REDES BAYESIANAS

Um passo importante no Algoritmo 1 está na primeira linha onde é criada uma solução inicial para o problema baseada em uma permutação dos atributos do conjunto de dados. O Algoritmo 3 mostra uma modificação do algoritmo anterior em que a solução inicial não é feita a partir de uma permutação.

Algoritmo 3: Melhora do algoritmo 1

```

1  L = BN_Initializer( x[ 0 ] , x[ 1 ] , ... , x[ n ] )
2  while k < NUM_RESTARTS :
3      best = find_order( L )
4      if score( best ) > score( L ) :
5          L = best

```

A implementação da função é mostrada no Algoritmo 4 e tem duas partes. A primeira parte é encontrar os melhores pais para cada um dos atributos do conjunto de dados. A segunda parte recebe os melhores pais e retorna um conjunto de arcos de realimentação (Feedback arcset, em inglês).

Algoritmo 4: Gerar uma solução inicial

```

1  for i in range( n ) :
2      pa_x[ i ] = findBestParents( x[ i ] )
3  return feedbackArcSet( pa_x[ 0 ] , ... pa_x[ n - 1 ] )

```

A ideia geral da segunda parte é que dados os pais para cada atributo e tendo todas as conexões entre os atributos não se pode formar um grafo sem ciclos. Então o problema é mudar de um grafo com ciclos para um sem ciclos (DAG, em inglês) o que é conhecido como o conjunto de arcos de realimentação ou Feedback Arcset (FAS).

Encontrar o FAS de um grafo é um problema NP-Hard, mas existem vários métodos de aproximação. O método implementado para este artigo é descrito a continuação:

- (1) Elegir um nó do grafo aleatoriamente
- (2) Faça uma busca em profundidade desde esse nó e marque todos os arcos visitados que não fazem ciclos
- (3) Todos os arcos do grafo que não foram marcados são invertidos
- (4) Retornar o novo grafo

Na seguinte seção este algoritmo será comparado com o algoritmo original na seção 2 usando vários conjuntos de dados.

4. EXPERIMENTOS

5. CONCLUSÕES

REFERENCES