Revisando Redes Bayesianas através da Introdução de Variáveis Não-observadas

Kate Revoredo[1] **, Aline Paes**[1] **, Gerson Zaverucha**[1] **, Vitor Santos Costa**[2]

[1]Departamento de Engenharia de Sistemas e Computação - COPPE
Universidade Federal do Rio de Janeiro (UFRJ)

[2]Departamento de Ciência de Computadores - Universidade do Porto

{kate,ampaes,gerson}@cos.ufrj.br, vsc@dcc.fc.up.pt

Abstract. *An important issue on learning Bayesian networks is how to effectively learn their structure in the presence of hidden variables. In this paper, we propose a new approach based on theory revision. Our algorithm searches for the best place to introduce a hidden variable guided by the examples and through the use of a discriminative approach. The hidden variables are included through a revision operator defined in this paper. Moreover, our algorithm is capable of introducing as many hidden variables as necessary to improve the performance of the Bayesian network and it can be applied even on sparse Bayesian networks. We successfully evaluated our algorithm on 12 real datasets.*

Resumo. *Uma questão importante em aprendizado de redes Bayesianas (RB) é como aprender a estrutura da rede na presença de variáveis não-observadas. Neste artigo, propomos uma nova abordagem que utiliza técnicas de revisão de teoria. O algoritmo proposto, denominado DAHVI, aplica uma heurística discriminativa e a partir dos exemplos identifica pontos potenciais na RB à inclusão de uma variável não-observada. Essas variáveis são incluídas através de um operador de revisão proposto neste artigo. O DAHVI é capaz de introduzir tantas variáveis não-observadas quantas forem necessárias para encontrar uma RB com uma melhor avaliação, podendo ser aplicado também a RB esparsas. Avaliamos com sucesso o DAHVI em 12 datasets reais.*

1. Introdução

Redes Bayesianas (RB) são grafos direcionados, onde cada nó é uma variável aleatória. Para construir uma RB, é preciso definir que variáveis são de interesse, como elas são condicionalmente independentes, isto é, a estrutura da rede, e os parâmetros das distribuições de probabilidade (CPDs) para cada variável aleatória da rede. Vários algoritmos de aprendizado de RB a partir de bases de dados, já foram propostos na literatura [Heckerman 1995]. Esses algoritmos vão desde o caso mais simples, onde a estrutura da rede é conhecida e os dados observados, apenas precisando aprender as CPDs, até casos mais complexos, onde os dados são parcialmente observados, a estrutura pode ou

não ser conhecida e variáveis aleatórias de interesse podem nunca terem sido observadas. Tais variáveis são conhecidas como *variáveis não-observadas*. Essas variáveis são importantes tanto para RB dinâmicas quanto estáticas. No último caso, geralmente funcionam como mecanismos de agrupamento, que capturam informações a partir de um conjunto de variáveis observadas passando essas informações para outra parte da rede. Dessa forma, a introdução de variáveis não-observadas pode simplificar a estrutura da rede. Uma estrutura concisa permite inferência e aprendizado melhores, já que, reduz o número de parâmetros e ainda diminui a chance de *overfitting*. A Figura 1 exibe um exemplo motivador, originalmente descrito em [Binder et al. 1997], onde H é uma variável não-observada que ao ser introduzida reduz o número de arestas na rede de 12 para 6, e, assumindo um domínio binário para cada variável aleatória, reduz o número de parâmetros probabilísticos em 61%.

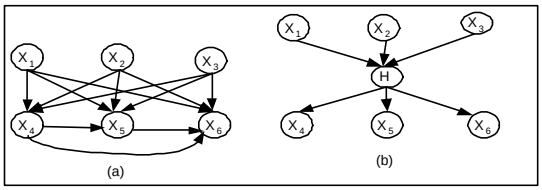


Figure 1. (b) Uma estrutura de RB com uma variável não-observada (H). (a) Uma estrutura de RB com a mesma distribuição sem a variável não-observada.

Infelizmente, aprender novas variáveis não-observadas em uma RB estática não é uma tarefa simples. Para cada nova variável não-observada é preciso aprender uma nova estrutura de rede e as CPDs. Existem pesquisas para minimizar esse problema. O algoritmo *Structural Expectation Maximization* (SEM) [Friedman 1998] combina aprendizado de estrutura com o algoritmo EM [Dempster et al. 1977, Lauritzen 1995] para RB, introduzindo variáveis não-observadas e aprendendo suas CPDs, mas exige que um conhecimento preliminar sobre a localização e número dessas variáveis não-observadas seja fornecido, para que ele consiga ter uma boa performance. Se não forem fornecidas tais informações, outras abordagens existem. O algoritmo *Semi-clique*[Elidan et al. 2000], por exemplo, considera que geralmente variáveis não-observadas podem simplificar segmentos da estrutura da rede que estão altamente conectados. Alternativamente, pode-se tirar vantagem da topologia da rede: BANNER [Ramachandran and Mooney 1998] introduz variáveis não-observadas em uma RB *noisy-or* e *noisy-and*.

Neste trabalho, introduzimos uma nova abordagem para aprender variáveis não-observadas baseada em revisão de teoria [Wrobel 1996, Buntine 1991]. Nosso algoritmo de revisão, denominado **DAHVI** (**D***iscriminative* **A***pproach for* **H***idden* **V***ariable* **I***ntroduction*), utiliza uma abordagem discriminativa: as variáveis de consulta e as suas Coberturas de Markov [1] correspondentes são os pontos de revisão. Nos baseamos na observação de que geralmente queremos melhorar a performance para um conjunto de

[1]Cobertura de Markov de um nó em uma RB consiste dos seus nós pais, dos seus nós filho e dos nós pai dos seus nós filho.

variáveis de consulta. Para isto, focamos nas suas Coberturas de Markov, já que as variáveis que influenciam o valor inferido para as variáveis de consulta pertencem a ela. Um operador de revisão, para especificamente introduzir variáveis não-observadas, é proposto neste artigo. Dessa forma, nosso algoritmo prossegue olhando para os exemplos inferidos incorretamente, computando as Coberturas de Markov para as variáveis de consulta, propondo variáveis não-observadas e avaliando se elas melhoram a performance da rede ou não. Aplicamos o algoritmo DAHVI a 12 datasets reais. Os resultados mostraram os benefícios do nosso algoritmo, inclusive, demonstrando que (i) a nossa abordagem para selecionar pontos de revisão reduz o espaço de busca, e (ii) o nosso operador de revisão introduz variáveis não-observadas em lugares importantes da RB, indicando ser uma boa heurística para definição da localização das variáveis não-observadas.

Este artigo esta organizado da seguinte forma: na Seção 2 revemos brevemente aprendizado de uma RB. Na Seção 3 nosso algoritmo é descrito. Os resultados experimentais são discutidos na Seção 4 seguidos pelas conclusões na Seção 5.

2. Aprendizado de Redes Bayesianas

Considere o conjunto finito de variáveis aleatórias discretas $Vars = \{X_1, ..., X_n\}$. Uma RB é um grafo direcionado acíclico, onde os nós correspondem às variáveis aleatórias $Vars$ e estas têm uma CPD associada, que quantifica o efeito dos pais em um determinado nó ($P(X_i|Pa(X_i))$, onde $Pa(X_i)$ denota os pais de X_i). A RB especifica uma distribuição de probabilidades conjunta sobre $Vars$ dada por: $P(X_1, ..., X_n) = \prod_{i=1}^{n} P(X_i|Pa(X_i))$.

Considere o conjunto de treinamento $D = \{d^1, ..., d^m\}$ onde cada exemplo d^i associa um valor a alguma (ou toda) variável em $Vars$. A tarefa de aprender a estrutura da RB busca pela melhor RB que descreve D. Uma abordagem usual é avaliar cada RB existente no espaço de busca de acordo com uma função de avaliação escolhendo a de melhor avaliação. Geralmente, um algoritmo subida de encosta guloso com recomeços aleatórios é utilizado, com um arco sendo inserido/excluído/alterado por vez.

Quando o conjunto de treinamento é incompleto, isto é, algum d^i não associa valores para alguma (ou toda) variável em $Vars$, a tarefa de aprendizado é computacionalmente e conceitualmente muito mais difícil. No caso de uma estrutura de rede fixa, os algoritmos EM [Dempster et al. 1977, Lauritzen 1995] ou gradient ascent [Binder et al. 1997] podem ser utilizados para aprender as CPDs. Se a estrutura não é conhecida, algoritmos tais como SEM [Friedman 1998] podem ser utilizados. Este algoritmo combina um passo EM com um passo de aprendizado de estrutura.

3. Uma abordagem discriminativa para introduzir variáveis não-observadas

Neste artigo, propomos um algoritmo de revisão de RB que introduz variáveis não-observadas. A motivação para o uso desta abordagem é a seguinte. Se após o processo de aprendizado, a RB resultante não foi capaz de inferir o valor correto de pelo menos uma

variável de consulta, considerando um determinado exemplo, então é possível que exista alguma informação "escondida" que não esteja sendo descrita corretamente pela estrutura de rede. Nosso algoritmo baseia-se em revisão de teoria [Wrobel 1996] da seguinte forma: **(i)** utiliza uma abordagem discriminativa para selecionar pontos potenciais a modificação, denominados *pontos de revisão*; **(ii)** aplica um *operador de revisão*, definido neste trabalho, que introduz variáveis não-observadas nestes pontos de revisão; **(iii)** avalia as redes propostas e escolhe a melhor para o próximo passo.

3.1. Seleção dos Pontos de Revisão

Uma parte importante do algoritmo de revisão proposto neste trabalho é a definição dos pontos de revisão, que representaremos por MB^*. Nós consideramos uma abordagem discriminativa, onde as variáveis de consulta ($\mathbf{Y}$) e as variáveis pertencentes às Coberturas de Markov correspondentes ($MB^* = Y \cup coberturaMarkov(Y)$) são os pontos de revisão. Selecionamos então da seguinte forma os pontos de revisão (MB^*):

Definição 3.1 *Se o conjunto de exemplos inferidos incorretamente, D_{mis}, é totalmente observado, então $MB^* = Y \cup coberturaMarkov(Y)$. Caso contrário,*

$$MB^* = \bigcup_{d_{mis_i} \in D_{mis}} MB^*_{d_{mis_i}}$$

onde $MB^*_{d_{mis_i}}$ ($MB^*_{d_{mis_i}} = MB^*_n$) é definido de forma recursiva a partir de $MB^*_0 = Y \cup coberturaMarkov(Y)$ considerando o exemplo d_{mis_i}:

$$MB^*_n = MB^*_{n-1} \bigcup coberturaMarkov(Var_{mis}(MB^*_{n-1}))$$

onde $Var_{mis}(MB^*_{n-1})$ é o conjunto de variáveis com valores não observados em d_{mis_i} considerando a iteração anterior.

A segunda parte da definição é dependente dos exemplos, já que as variáveis que têm informação perdida podem variar de exemplo para exemplo. Para cada exemplo inferido incorretamente (d_{mis_i}), $MB^*_{d_{mis_i}}$ é definido de forma recursiva. A cada chamada recursiva, $MB^*_{d_{mis_i}}$ é acrescido da Cobertura de Markov de variáveis cujo os valores não foram observados no exemplo d_{mis_i} e que tenham sido inseridas em $MB^*_{d_{mis_i}}$ na iteração anterior. A recursão é finalizada quando não forem identificadas novas variáveis na iteração anterior. Para ilustrar esta definição, considere a RB exibida na Figura 2 (a) e os 4 exemplos para esta rede descritos a seguir, cujo valor para a variável de consulta Y não foi inferido corretamente. O respectivo $MB^*_{d_{mis_i}}$ é exibido para cada exemplo.

$d_{mis_1} = [y_1, x_{3_1}, x_{1_1}, x_{2_1}, x_{4_1}] \longrightarrow MB^*_{d_{mis_1}} = \{Y, X_3\}$
$d_{mis_2} = [y_2, x_{1_2}, x_{2_2}, x_{4_2}] \longrightarrow MB^*_{d_{mis_2}} = \{Y, X_3, X_1, X_2\}$
$d_{mis_3} = [y_3, x_{1_3}, x_{4_3}] \longrightarrow MB^*_{d_{mis_3}} = \{Y, X_3, X_1, X_2, X_4\}$
$d_{mis_4} = [y_4, x_{3_4}, x_{1_4}, x_{4_4}] \longrightarrow MB^*_{d_{mis_4}} = \{Y, X_3\}$

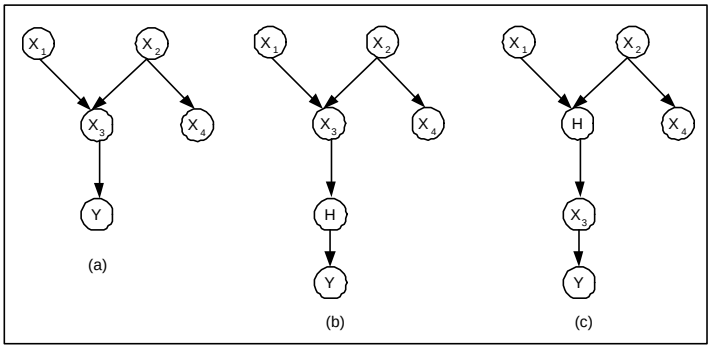


Figure 2. (a) RB, onde Y é a variável de consulta. (b) RB candidata considerando o nó $Y \in MB^*$ (c) RB candidata considerando o nó $X_3 \in MB^*$

O exemplo d_{mis_1}, por exemplo, fornece valor para todas as variáveis da rede, logo é totalmente observado, recaindo na primeira parte da Definição 3.1. Com isso, os pontos de revisão são a variável de consulta Y e a sua Cobertura de Markov composta pela variável X_3, definindo $MB^*_{d_{mis_1}} = \{Y, X_3\}$. O exemplo d_{mis_4}, apesar de ser parcialmente observado, pois não tem informações a respeito de X_2, também vai retornar $MB^*_{d_{mis_4}} = \{Y, X_3\}$, já que todas as variáveis pertencentes ao conjunto encontrado inicialmente ($MB^*_0 = Y \cup coberturaMarkov(Y) = \{Y, X_3\}$) têm informação e com isso não é necessário buscar pelas Coberturas de Markov correspondentes. Já os exemplos d_{mis_2} e d_{mis_3} são parcialmente observados e variáveis com informações perdidas são consideradas em MB^*_n, para algum $n \in N$, logo é preciso buscar a Cobertura de Markov destas variáveis. Em d_{mis_2} buscamos por $coberturaMarkov(X_3)$ e em d_{mis_3} por $coberturaMarkov(X_3)$ e $coberturaMarkov(X_2)$. O conjunto final dos pontos de revisão é a união dos $MB^*_{d_{mis_i}}$, ou seja $MB^* = \{Y, X_3, X_1, X_2, X_4\}$. Note que no pior caso, como neste exemplo, o conjunto de pontos de revisão inclui todas as variáveis. Este seria o caso para RB construídas por *Naive Bayes* e por TAN (*Tree-augmented network*) [Friedman et al. 1997], como em ambos a Cobertura de Markov da variável de consulta inclui todas as variáveis. O nosso algoritmo funciona melhor com redes menos conectadas, construídas por algoritmos como K2 ou SEM [Friedman 1998].

3.2. Operador de Revisão

O operador de revisão proposto neste artigo, define uma estrutura candidata com uma variável não-observada intermediando uma ponto de revisão e seus pais. Cada variável não-observada é introduzida considerando cardinalidade 2, ou seja um domínio binário, já que assim aumentamos o número de parâmetros probabilísticos o mínimo possível. Em seguida, as CPDs são aprendidas, utilizando o algoritmo EM, e a RB candidata é avaliada utilizando uma função de avaliação probabilística.

No exemplo da seção 3.1 foram selecionados cinco pontos de revisão ($MB^* = \{Y, X_3, X_1, X_2, X_4\}$). Para cada um deles, o operador de revisão propõe uma RB candidata com uma variável não-observada incluída. Por exemplo, na Figura 2 (b) e (c)

uma estrutura candidata para os pontos de revisão Y e X_3 respectivamente é exibida: na primeira, a variável não-observada H é introduzida intermediando a variável de consulta Y e seus pais e no segundo exemplo H é introduzida entre X_3 e seus pais. Apesar de apenas acrescentar uma variável não-observada intermediária entre 2 variáveis, a avaliação das redes candidatas pode ser alterada para melhor, já que as CPDs são outras. Além disso, dependendo do tamanho do domínio das variáveis envolvidas, ao acrescentar a variável não-observada reduz-se o número de parâmetros probabilísticos, podendo tornar a inferência e o aprendizado dos parâmetros mais rápido e preciso.

3.3. Algoritmo de Revisão DAHVI

O algoritmo DAHVI exibido em Algoritmo 1, recebe uma RB, uma função de avaliação probabilística (*score*), as variáveis de consulta (**Y**) e o *dataset* (**D**). Quando o *dataset* considerado é de classificação, as variáveis de classe são as variáveis de consulta utilizadas pelo DAHVI para selecionar os pontos de revisão. Para *datasets* mais gerais, ou seja *datasets* que não são de classificação, o usuário define quais variáveis são mais adequadas para guiarem esta seleção. DAHVI começa avaliando a RB, recebida como entrada, e assume que RB é a melhor rede até o momento através da inicialização da variável RB_h, que será retornada ao final da execução do algoritmo (passos 1 e 2). Em seguida, no passo 4, os exemplos, onde pelo menos uma das variáveis de consulta não teve o seu valor inferido corretamente, são reunidos na variável D_{mis}. Caso D_{mis} seja vazio, o algoritmo termina já que não existe necessidade em revisá-lo (passos 5 e 6). Caso contrário, os pontos de revisão são selecionados como descrito na Seção 3.1 (passo 7). No passo 8, o nosso operador de revisão é aplicado em cada ponto de revisão, como definido na Seção 3.2. A melhor estrutura candidata é escolhida no passo 9 de acordo com a avaliação. Se a melhor RB candidata melhorar a RB corrente, ela é implementada e o algoritmo prossegue (passos 10 e 11). O algoritmo é subida de encosta, logo é executado enquanto a estrutura de rede puder ser melhorada. Ao final a rede RB_h é retornada.

Algoritmo 1 Algoritmo **DAHVI**

Entrada: RB; função de avaliação (*score*); variáveis de consulta (**Y**); *dataset* (**D**);
Saída: uma rede Bayesiana com variáveis não-observadas (RB_h)

1: $RB_h = RB$;
2: **repete**
3: $Score_h = score(RB_h, \mathbf{D})$;
4: D_{mis} = exemplos inferidos incorretamente;
5: **se** $D_{mis} = \varnothing$ **então**
6: break;
7: encontre MB^* como na Definição 3.1;
8: encontre o conjunto de estruturas candidatas a partir de MB^*;
9: escolha a melhor estrutura candidata BN_{best} a partir da sua avaliação ($Score_{melhor}$)
10: **se** $Score_{melhor} > Score_h$ **então**
11: $RB_h = RB_{melhor}$;
12: **até** a avaliação convergir

DAHVI inclui tantas variáveis não-observadas quantas forem necessárias para

melhorar a avaliação da RB. Diferentemente do algoritmo SEM, onde é necessário indicar a priori quantas são as variáveis não-observadas, DAHVI descobre quantas são executando o algoritmo. Além disso, DAHVI pode ser aplicado em redes esparsas.

4. Resultados Experimentais

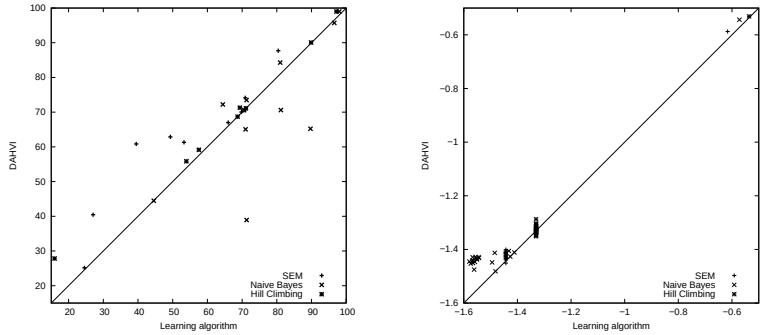
Conduzimos experimentos em *datasets* discretos com características diversas: (i) *datasets* de classificação ou gerais; (ii) totalmente ou parcialmente observados; (iii) com muitos ou poucos exemplos. Dessa forma, escolhemos aleatoriamente 10 *datasets* de classificação no repositório do UCI (Audiology, Breast Cancer, Breast Cancer Wisconsin, Car evaluation, Lymphograph, Nursery, Post-operative Patient, Primary Tumor, Tic-tac-toe Endgame and Zoo) e 2 *datasets* gerais utilizados anteriormente para avaliar a introdução de variáveis não-observadas [Elidan et al. 2000]:

a) *Stock* - mostra o desempenho de 20 empresas americanas de tecnologia na bolsa de valores, indicando se suas ações subiram, desceram ou mantiveram-se estáveis durante 1516 dias.
b) *Tuberculose* (TB) - contém informações demográficas e médicas sobre 2302 pessoas com tuberculose em um bairro de São Francisco, USA.

Utilizamos 10-fold validação cruzada, separando os dados em conjuntos disjuntos de treinamento e teste. Além disso, em cada conjunto de treinamento,para evitar overfitting, foi utilizado 5-fold validação cruzada, separando este conjunto em conjuntos disjuntos de treinamento e validação [Kohavi 1995], onde a melhor RB, considerando o conjunto de validação, é avaliada no conjunto de teste. Avaliamos as RB considerando a média da log-verossimilhança (LL), nos *datasets* gerais, e da acurácia (ACC), nos *datasets* de classificação, das variáveis de consulta. Com o intuito de comprar diferentes *datasets*, as avaliações computadas foram normalizadas pelo tamanho da base de dado correspondente. Utilizamos t-test corrigido [Nadeau and Bengio 2003] com 95% de confiança para verificar a significância das diferenças entre as médias.

O primeiro experimento tem por objetivo mostrar que a revisão de uma RB através da inclusão de variáveis não-observadas melhora a sua avaliação. Dessa forma, uma RB é aprendida utilizando um algoritmo de aprendizado conhecido: neste artigo, utilizamos SEM, *hill-climbing* (HC) e *Naive Bayes* (NB). Não foi possível encontrar uma rede HC para o *dataset Audiology*, devido ao número de atributos e ao tamanho do domínio destes. Além disso, não esperamos grandes benefícios quando aplicando DAHVI as redes NB, já que a cobertura de Markov neste caso é toda a rede, e o NB beneficia-se da sua estrutura simples. Em seguida, revisamos as redes aprendidas utilizando o DAHVI. Espera-se que a RB revisada tenha uma avaliação melhor nos dados de teste do que a RB aprendida.

Para os 10 *datasets* de classificação, utilizamos log-verossimilhança condicional (CLL) como função de avaliação, já que [Grossman and Domingos 2004] mostrou que esta função é mais apropriada para a tarefa de classificação. A Figura 3(a) mostra o gráfico de dispersão comparando a média da ACC para as redes revisadas pelo DAHVI



(a) Datasets de classificação (b) Datasets gerais

Figure 3. (a) ACC do DAHVI versus ACC do SEM,HC e NB, considerando CLL como função de avaliação para os 10 datasets de classificação. (b) LL do DAHVI versus LL do SEM, HC e NB considerando LL como função de avaliação e os 2 datasets gerais.

com as redes aprendidas. DAHVI melhorou significativamente a ACC da rede SEM em metade dos *datasets*, não piorando em nenhum. Já revisando as redes HC, DAHVI obtém melhoras, mas não tantas quanto as obtidas com as redes SEM. A pior performance foi para as redes NB, como esperado. Para os *datasets* TB e *Stock*, como estamos interessados em melhorar o modelo considerando toda a rede, utilizamos LL como função de avaliação. Como nossa abordagem requer uma variável de consulta, escolhemos o tipo de tuberculose para o *dataset* TB e rodamos o algoritmo DAHVI 20 vezes, em cada rodada considerando uma das 20 variáveis como sendo a variável de consulta, já que a probabilidade de escolha de uma entre as 20 variáveis é uniforme. O gráfico de dispersão exibido em 3(b) compara a LL retornada pelos sistemas SEM, HC e NB com a LL retornada pelo algoritmo DAHVI. Este obteve RB melhores considerando todos os 3 algoritmos de aprendizado quando aplicados o *dataset* TB e considerando o *dataset* Stock ele melhorou a LL da RB aprendida utilizando HC em 11 das 20 variáveis, 19 das 20 variáveis usando SEM e 17 das 20 variáveis usando NB. Dessa forma, verificamos que a nossa abordagem é aplicável a *datasets* gerais. Verificamos também que o DAHVI reduziu em mais de 50% o espaço de busca considerado para inclusão de variáveis não-observadas em todos os *datasets* quando revisando as redes HC e SEM, com exceção do Nursery onde a redução foi de 33.34%, quando revisando a rede HC.

O segundo experimento tem por objetivo verificar se a nossa abordagem para seleção dos pontos de revisão é válida, ou seja se a heurística para identificação de onde a variável não-observada deve ser inserida é boa. Comparamos então os resultados obtidos pelo DAHVI, com os obtidos pelo SEM, onde esse recebeu a informação de que era preciso incluir variáveis não-observadas (SEM_h) e com o algoritmo *Semi-clique* [Elidan et al. 2000]. Como a média de variáveis não-observadas inseridas pelo DAHVI ficou entre 1 e 2, consideramos que no geral 1 variável não-observada foi intro-

duzida e rodamos o SEM indicando para ele que 1 variável não-observada precisava ser inserida. Quando comparado com o SEM_h, DAHVI obteve melhores resultados em 8 dos 12 *datasets*. Além disso, o SEM_h na maioria das vezes não melhorou a performance da rede quando comparado com SEM, diferente do DAHVI como vimos no primeiro experimento. A comparação com o algoritmo *Semi-clique*, que procura por sub-estruturas da rede aprendida, denominadas semi-cliques, as quais sugerem a presença de uma variável não-observada, só foi possível para as redes HC e SEM, já que as redes NB não possuem semi-cliques e assim o algoritmo *Semi-clique* não inclui nenhuma variável não-observada. Apenas para as bases Nursery (i) e Stock (ii) as redes SEM apresentaram *semi-cliques* e o algoritmo *Semi-clique* pode ser aplicado. Os resultados obtidos mostram que o DAHVI obteve uma LL melhor: (i) -1.0492 e -1.0574; (ii) $\mathbf{-1.4008}$ e -1.5362 (os valores em negrito indicam que a diferença é estatisticamente significativa). Já o algoritmo HC aprendeu redes com cliques em 4 *datasets*, Nursery (i), Stock (ii), Zoo (iii), and TB (iv). Para estes o DAHVI também obteve melhores resultados: (i)-0.9686e -0.9828 (ii) $\mathbf{-1.2875}$ e -1.3395 (iii) $\mathbf{-1.0200}$ e -1.4038; (iv) $\mathbf{-0.5311}$ e -0.5816. Além disso, em todos os casos o *Semi-clique* não melhorou a LL da RB inicial no conjunto de teste.

5. Conclusão

Uma questão importante, mas desafiadora em aprendizado de RB é como aprender a estrutura da rede na presença de variáveis não-observadas. Neste trabalho, introduzimos uma nova abordagem para aprender RB com variáveis não-observadas, baseado em revisão de teoria [Wrobel 1996]. Descrevemos um algoritmo, chamado DAHVI, o qual verifica se uma RB pode ser melhorada através da inclusão de variáveis não-observadas nos pontos de revisão. Os resultados experimentais mostraram que o DAHVI é capaz de melhorar uma RB aprendida a partir de diferentes algoritmos de aprendizado disponíveis na literatura, encontrando os melhores resultados quando revisando a RB aprendida com o algoritmo SEM. Os experimentos também mostraram que o DAHVI seleciona o melhor número de variáveis não-observadas durante a execução do algoritmo, diferentemente do sistema SEM, que precisa desta informação a priori. Uma possível extensão do algoritmo DAHVI é a consideração de diferentes domínios para as variáveis não-observadas, além do binário [Elidan and Friedman 2005] e a definição de outros operadores de revisão que propõem outras modificações além da introdução de variáveis não-observadas.

Agradecimentos

O primeiro e o terceiro autores são parcialmente financiados pelo CNPq e o segundo pelo CNPq e CAPES.

References

Binder, J., Koller, D., Russell, S., and Kanazawa, K. (1997). Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29:213–244.

Buntine, W. (1991). Theory refinement on bayesian networks. In *Proceedings Seventeenth Conference Uncertainty in Artificial Intelligence*, pages 52–60, San Mateo, CA.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Royal Stat Soc*, 39:1–39.

Elidan, G. and Friedman, N. (2005). Learning hidden variable networks: The information bottleneck approach. *Journal of Machine Learning Research*, 6:81–127.

Elidan, G., Lotner, N., Friedman, N., and Koller, D. (2000). Discovering hidden variables: a structure-based approach. In *Neural Information Processing Systems*, volume 13, pages 479–485.

Friedman, N. (1998). The bayesian structural EM algorithm. In *UAI*, pages 129–138.

Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, 29:131–163.

Grossman, D. and Domingos, P. (2004). Learning bayesian network classifiers by maximizing conditional likelihood. In *Proc. 21th Int. Conference on Machine Learning*, pages 361–368.

Heckerman, D. (1995). A tutorial on learning bayesian networks. Technical report, Microsoft Research.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the International Joint Conference on Artificial Intelligence(IJCAI)*, pages 1137–1145.

Lauritzen, S. L. (1995). The em algorithm for graphical association models with missing data. *Comp. Stat.and Data Ana.*, 19:191–201.

Nadeau, C. and Bengio, Y. (2003). Inference for the generalization error. *Machine Learning*, 52(3):239–281.

Ramachandran, S. and Mooney, R. (1998). Theory refinement of bayesian networks with hidden variables. In *Proc. 15th Int. Conference on Machine Learning*, pages 454–462.

Wrobel, S. (1996). First-order theory refinement. *Advances in Inductive Logic Programming*, pages 14–33.