Information Sciences 439–440 (2018) 108–124

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Revising the structure of Bayesian network classifiers in the presence of missing data

Roosevelt Sardinha[a], Aline Paes[b,*], Gerson Zaverucha[a]

[a] Department of Systems Engineering and Computer Science, COPPE, Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro, RJ, Brazil

[b] Department of Computer Science, Universidade Federal Fluminense (UFF), Niterói, RJ, Brazil

ARTICLE INFO

Article history:
Received 2 March 2017
Revised 5 February 2018
Accepted 6 February 2018
Available online 8 February 2018

Keywords:
Refinement of Bayesian networks
Structure learning
Missing data
Bayesian network classifiers

ABSTRACT

Traditionally, algorithms that learn the structure of Bayesian Networks either start from an empty graph and add edges to it bit by bit or add/ remove/reverse edges in a randomly initialized graph. In both cases, the search space is constituted of all the nodes and edges connecting them. Searching within such a vast scope is a hard task, which gets worse in the presence of a dataset with missing values. However, it may be the case that an initial structure already exists and to make it reflect the set of examples it would be required to modify only a subset of the graph. Thus, instead of searching through the entire space of possible connections between the nodes, the problem could be reduced to selecting a subset of the edges and revising them. In this work, we present a novel algorithm for refining the structure of Bayesian networks from incomplete data, named BaBReN (Bayes Ball for Revising Networks). BaBReN has as ultimate goal to improve the inference value of the class variable. Thus, the algorithm tries to solve classification issues by proposing local modifications to the edges connecting the nodes that influence the erroneous classification. The Bayes Ball algorithm – based on the d-separation criteria – is responsible for selecting those relevant nodes. By focusing only on the influential nodes, BaBReN is executed independently of the number of variables in the domain. BaBReN is compared to a constraint-based algorithm (GS), a hybrid one (MMHC) and a score-based one (SEM with GHC), presenting better or competitive results regarding time and classification score.

© 2018 Elsevier Inc. All rights reserved.

1. Introduction

A Bayesian network (BN) is a probabilistic graphical model that compactly represents a joint distribution of random variables [35]. This compact and elegant representation is achieved by associating a node to each random variable in a directed acyclic graph (DAG). In this DAG, an edge connecting a pair of nodes stands for a direct relationship between those nodes. Thus, Bayesian networks can represent the relationship between variables in a human-readable way, while still taking into account local probability distributions and (conditional) independence. Because of that, one can use a Bayesian network to answer questions concerning any variable encompassed in this distribution.

Bayesian networks have been widely used to represent noisy data and perform uncertain reasoning in a great variety of classification and prediction problems. These include genetic linkage analysis, bio-medicine, health-care, decision sup-

* Corresponding author.
E-mail addresses: roosevelt@cos.ufrj.br (R. Sardinha), alinepaes@ic.uff.br (A. Paes), gerson@cos.ufrj.br (G. Zaverucha).

https://doi.org/10.1016/j.ins.2018.02.011
0020-0255/© 2018 Elsevier Inc. All rights reserved.

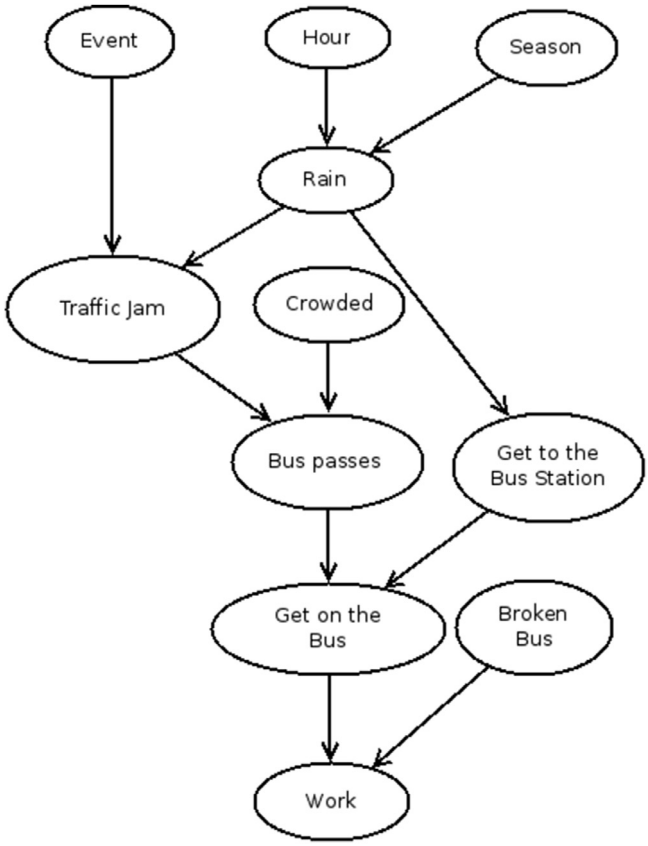

Fig. 1. An example of the structural component (the DAG) of a Bayesian network.

port on crime risk factors, speech recognition, information theory, medical diagnosis, troubleshooting, data mining and pattern recognition, sensor validation algorithms, information retrieval, reliability analysis of safety-critical systems, among others [13,35].

The DAG depicted in the Fig. 1 is the *structure* of a BN. Together with a set of *probability parameters*, it is possible to compute the probability of a variable assuming a certain value, given the values of the other variables, or still to find the most probable value of a group of variables. For instance, one could obtain the answer to the following questions: *"what is the probability of the variable* ***Rain****to assume the value* ***little****, given that the variable* ***Season****has assumed the value* ***summer****and the variable* ***Traffic Jam****has assumed the value* ***low****"?*; or, *"Which one is the most probable value of the variable* ***Get to the Bus Station****, when variables* ***Rain, Crowded****, and* ***Bus passes****have assumed values* ***little, much****, and* ***yes****, respectively"?* Several inference algorithms have been designed to compute the probability for such types of queries. Examples include exact inference methods like the *Variable Elimination* and the *Junction Tree* and approximate methods based on the Markov Chain Monte Carlo (MCMC) strategy, like *Gibbs Sampling* [26].

The structure and the probability parameters of a BN might be fully elicited by an expert of the domain. However, this is usually a complicated, prone to error and time-consuming task. Furthermore, it may lead to too specific networks, since they would be built from the opinion and intuition of the specialists involved. Thus, when there is available data, a better approach is to automatically *learn* the network structure and parameters. A set of algorithms have been developed for those tasks, such as the methods based on Gradient Ascent and Expectation Maximization [26], to learn the probability parameters, and *Grow and Shrink* [30], *K2* [12] *Max-Min Hill Climbing* [45], *Greedy Equivalent Search* [31], *Dynamic Markov Blanket Classifier* [39], B&B [8], among others, to learn the structure.

Most of the aforementioned structure learning algorithms require complete data to operate. However, real-world data usually have missing values, due to failure on sensors, human shortcomings, or even as a consequence of the behavior of other variables in the domain. Thus, in the presence of missing data, one may have to resort to ad-hoc methods that complete the values of the variables, which may lead to biased results and wrongly indicate overconfidence in their analysis. There are only a few alternatives to learn the structure with incomplete data, including algorithms like the Structural Expectation Maximization (SEM) [18] and its variants. However, because of the high number of inferences accomplished when working with incomplete data, these approaches still take too much time to run.

Download English Version:

https://daneshyari.com/en/article/6856578

Download Persian Version:

https://daneshyari.com/article/6856578

Daneshyari.com