Probabilistic Models Project Proposal

Final Project

Computer Science
IDC
February 20, 2017

Overview
Create a tool for students to both learn and experiment in probabilistic models. The idea is to illustrate different aspects of probabilistic models, all is defined by the units presented in the scope section. The tool is design to help from the basic understanding of dependencies for random variables, to running different algorithms on Bayesian networks. This file is going to explain the scope and way to create a tool for students to run various probabilistic models described in this document. The tools is going to be assembled from the following components:

1. Visual representation of model
2. JSON representation for models
3. CLI command tool to run operation against a model and to see reports

Network Definition
In the visualization part the user will have the ability to:

1. Define Bayesian network in a couple of steps:
   (a) Define all the different nodes. Where every node have a name and domain.
   (b) Define the dependencies between the nodes.

   At the end of this process the user is able to see a graph representing the network.
2. Run different algorithms on a Bayesian network such as elimination, message passing. And in addition run transformations such as to moralized graph

Define Conditional Table The user is able to add conditional table for the nodes in an easy way. The user will be able to choose a node in the network defined in network definition table, and the node will be highlighted and an empty conditional distribution table will be presented, where the user is able to set the probability for every value of the node given his parents $P(X_v|X_{\pi_v})$

Scope The tool is going to build from four units, where every unit will extend the capabilities of the former one:

1. Unit 1
   This unit is the designed to learn the basics of probability modeling of data. After understanding the basics in probability, visualize dependency in discrete random variables, and how those variables can be dependent or independent on other random variables.
   (a) Represents dependencies through directed graph. Create predefined examples of models to show dependency vs independence of RVs.
   The user will be able to create dependency in the steps defined in network definition section, the UI will represent the dependency hierarchy.
   (b) Represent Bayesian network through an directed acyclic graph. To see how the definition is done see network definition section
   The graph representation is giving the user the ability to better understand the network with conditional table. To see how to add conditional table go to define conditional table section
   (c) show conditional independence using D separation rules. The rules are:
      i. Path contains a node $w \in A$ and not both adages that touch w are incoming:
         A. $X_u \to x \to X_v$
         B. $X_u \leftarrow x \leftarrow X_v$
         C. $X_u \leftarrow x \to X_v$
      ii. path contains a node $w \notin A$ with two incoming edges $X_u \to w \leftarrow X_v$

   The D separation is a general criterion for $X_v \perp\!\!\!\perp X_u|X_A$. Where X_A is a set of RVs need to condition on in order that v will be independent from u. The user is going to be able to choose two nodes, the tool is going to show id those nodes are depended or not using D separation. If they are conditional independent show the user all the paths that does not block dependency, otherwise show the path that block dependency.

2. Unit 2

   (a) Covert graph representation of a model to moralized graph. Validate if graph is chordal or not

   (b) Convert moralized graph to factor graph

3. Unit 3

   (a) Show elimination algorithm implementation on graphs

   (b) Find perfect elimination for tree. The perfect elimination can be found using the moralized graph from unit 2, validate if the graph is chordal or not.

   (c) Show message passing algorithm in graph representation. Display the marginal distribution tables of vertexes along the algorithm.

4. Unit 4

   (a) Parameter inference through data in tree models

5. Unit 5

   (a) Markov chain Monte Carlo algorithm

Model Representation
A model is represented using:

1. Random variables. Where every random variable can have:

   (a) Name (X)

   (b) Domain Ω where every x in Ω have the probability $P(X = x)$

Example for JSON file, representation of random variable
variable:
- name: x1
  domain:
  - 0
  - 1
- name: x2
  domain:
  - 0
  - 1
  - 2

Visual Representation

1. Represent Bayesian network using DAG. Represent all the nodes, with the direct edges from parents to node $P(X_v|X_{\pi_v})$

2. Represent the elimination algorithm for a Bayesian network. $f(x_{hidden}) = P(x_{hidden}, X_{observed}) = \sum_{\forall x \in V} P(X_v|X_{\pi_v})$. This is designed to show the elimination algorithm step by step.
   Setup:

   (a) Set all the nodes data and with the observed nodes value.

   (b) set the elimination order

   Initialization:

   (a) Set potential function for all the nodes $\Psi_{v \in V}$, where $\psi_v = P(X_v|X_{\pi_v})$

   (b) create a matrix $|V \setminus observed|X|\Psi|$ of all RVs associate with every potential function.

   (c) Set the list of elimination order of RVs in $X_{V \setminus (observed \cup infered)}$

   Elimination Loop:

   (a) X_v is the next RV in the elimination order so Ψ_v is the potential functions have RV in the matrix

   (b) $n(X_v)$ are the RVs that involved in one of the potential function in Ψ_v

(c) set m_v of all RVs in $n(X_v)$. So $m_v(n(X_v)) = \sum_{X_v} \Psi_v$

(d) replace Ψ_v with m_v

By setting up elimination order the user is able see the impact on the complexity.
Output of the run - the elimination step by step with potential matrix.

3. Convert Bayesian network to moralized graph. In this scenario the objective is to show how the operation is done step by step.

4. Message passing algorithm. Visual representation of the message passing. Step by step running of eliminate procedure

5. Factor Tree and message passing algorithm. Represent the conversion between a direct graph to the moralized version and then to factor graph

CLI tool

1. Running elimination algorithm for given node setup with elimination order.
Command: eliminate "path to file" "elimination order for all the nodes"
Input:

   (a) JSON file path, representing the RVs, nodes and conditional tables for all nodes $P(X_v = x_v | X_{\pi_v} = x_{\pi_v})$.

   (b) elimination order of the nodes

   Output:
   Joint probability $f(\overline{x_{infered}}) = P(X_{infered}, X_{observed})$

2. Message passing algorithm in trees. Run only eliminate procedure. Command: message passing "sync/async" "path to file" "elimination order for all the nodes"
Give the ability to run in two modes:

   (a) synchronous - eliminate one child of a node at a time

   (b) asynchronous - eliminate child nodes in parallel

   Message passing algorithm with compute marginal for all nodes. Run the procedures collect and distribute

3. Message passing with most probable joint assignment. The input is the same as the other message passing CLI arguments. The output in this case is maximum probability and the assignment.

4. Run message algorithm on poly tree. Given as input:
Command: message passing "sync/async" "path to file" "elimination order for all the nodes"

   (a) poly tree with nodes V and edges E

   (b) conditional tables for all nodes $P(X_v = x_v | X_{\pi_v} = x_{\pi_v})$

   Output:
   Marginal distribution for all the nodes

Design