Learning Parameters of Chain Logic Theories

Arjen Hommersom ARJENH@CS.RU.NL
Nivea Ferreira NIVEA@CS.RU.NL
Peter Lucas PETERL@CS.RU.NL
Institute for Computing and Information Sciences, University of Nijmegen
Heyendaalseweg 135, 6525AJ Nijmegen, The Netherlands

Keywords: chain graphs, chain logic, learning

Abstract

There has been a considerable amount of research on statistical relational learning (SRL) during the past decade, much of it focusing on generalised probabilistic logics . SRL formalisms are typically based on Bayesian or Markov networks. Chain graphs (CGs) generalise both acyclic directed graph (ADG) models, i.e., Bayesian networks, and undirected graph (UG) models, i.e., Markov networks, as they may contain both undirected and directed edges. As a consequence they constitute an attractive point of departure for SRL. In this paper, we start with a logical generalisation of chain graphs, called *chain logic*, to elaborate a method for learning the parameters of chain logic theories.

1. Chain Logic

1.1. The Language

Chain logic is a variant of probabilistic Horn logic as introduced by Poole in (Poole, 1993) augmented with integrity constraints (Ferreira et al., 2008). The language consists of a special variant of function-free Horn logic, where the syntax of Horn clauses is slightly modified, and logical implication, '$\leftarrow$', is given a causal interpretation. The clauses have the following form:

$$D \leftarrow B_1, \ldots, B_n : R_1, \ldots, R_m$$

where the predicates of the atoms D and B_i are at least unary and the atoms R_j, called *templates*, express relationships among variables, where at least one variable appearing in the atoms D and B_i occurs in at least one template R_j. From a logical point of view, the ':' operator has the meaning of a conjunction; it is only included in the syntax to allow separating atoms that are templates from non-template atoms. In chain logic, the templates are interpreted as representing uncertain events. The actual definition of the uncertainty is done by means of a *weight* declaration. This is of the form

$$weight(a_1 : w_1, \ldots, a_n : w_n) \quad (1)$$

where a_i represents an atom and $w_i \in \mathbb{R}_0^+$. The set of (ground) atoms appearing in such declarations are here called the *assumables*, denoted by $\mathcal{A}$. All other ground atoms are called *observables*, denoted by $\mathcal{O}$. Furthermore, we define a *hypothesis* as a conjunction of assumables, each of them occurring exactly once in a grounded weight declaration. The set of all such hypotheses is called $\mathcal{H}$. The set of consistent hypotheses, with respect to a chain logic theory T will be denoted by CH, i.e., $\text{CH} = \{H \in \mathcal{H} \mid T \cup H \nvDash \bot\}$.

Reasoning can be done by means of abduction: given a set of observed atoms $O \subseteq \mathcal{O}$, then these observations are explained in terms of the theory and an hypothesis.

Definition 1. *An* explanation *of a set of atoms O based on the pair $\langle T, \mathcal{A}\rangle$ is defined as an hypothesis $H \in \mathcal{H}$ satisfying the following conditions:*

- *$T \cup H \vDash O$, and*
- *$T \cup H$ is consistent, i.e., $T \cup H \nvDash \bot$.*

A *minimal explanation* of O is an explanation whose proper subsets are not explanations of O. The set of all minimal explanations is denoted by $\mathcal{E}_T(O)$.

Assumptions are imposed on chain logic theories, such that the clauses are acyclic, covering, mutually ex-

Presented at *ILP-MLG-SRL*, Leuven, Belgium, 2009.

clusive, and consistent. We define a joint probability distribution over the assumables such that each assumable is independent, i.e., joint probabilities can be easily calculated by multiplication and subsequent normalisation. As a result, we have a probability distribution associated to hypotheses and explanations. Furthermore, under the assumptions mentioned above we derive a joint distribution P_T over the observables: if $\mathcal{E}_T(O)$ is the set of minimal explanations of the conjunction of atoms $O \subseteq \mathcal{O}$ from the chain logic theory T, then:

$$P_T(O) = \sum_{H \in \mathcal{E}_T(O)} P(H) \tag{2}$$

1.2. Representing Chain Graphs

A chain graph $G = (V, E)$ is a graph with arcs (directed edges) and lines (undirected edges), such that there is no cycle in that graph that includes at least one arc (Lauritzen, 1996). A *chain component* is a connected component if all the arcs are removed from the graph; the set of all chain components is denoted by $\mathcal{C}$. Associated to a chain graph is a joint probability distribution $P(X_V)$ that factorises according to

$$P(X_V) = \prod_{C \in \mathcal{C}} P(X_C \mid X_{\mathrm{pa}(C)}) \tag{3}$$

where $pa(C)$ denotes the set of parents of C. Each $P(X_C \mid X_{\mathrm{pa}(C)})$ factorises according to

$$P(X_C \mid X_{\mathrm{pa}(C)}) = Z^{-1} \prod_{M \in M(C)} \varphi_M(X_M) \tag{4}$$

where Z^{-1} is a normalising constant and $M(C)$ is the complete set in the moral graph obtained from the subgraph $G_{C \cup \mathrm{pa}(C)}$ of G. The functions φ are real positive functions, called *potentials*, defined over cliques, i.e., maximal complete subgraphs of C; they generalise joint probability distributions in the sense that they need not be normalised. Finally, given two cliques, K and K', in an undirected graph, we call S a separator set iff $S = K \cap K'$.

Reasoning in chain logic can be identified with reasoning in the associated CGs: let T be a chain logic theory with ground atoms $V_i(c_i)$ and let v_i be the vertices in the corresponding (discrete) CGs, then:

$$P_T(V_1(c_1), \ldots, V_n(c_n)) = P(X_{v_1} = c_1, \ldots, X_{v_n} = c_n)$$

2. Learning Chain Graph Parameters

Where observables and assumables make up the core of chain logic, determining the probabilistic parameters of assumables using observations stored in a database D is one of the essential task of learning in chain logic.

In general, it is not possible to easily estimate the potentials from data as they might have a complex dependency to the rest of the graph. However, if the individual components are triangulated, the factorisation can be stated in terms of marginal probabilities over the variables in a clique.

The proposed algorithm for determining the parameters is inspired by the use of a junction tree for probabilistic inference as junction trees provide sufficient information about the interactions between assumables, i.e., when they influence the same observables. Junctions trees, with required properties such as the running intersection property, are only guaranteed to exist when the graph is triangulated, so we restrict ourselves to this case. As a convenience we write N_O with $O \subseteq \mathcal{O}$ for the number of tuples of D that contain O.

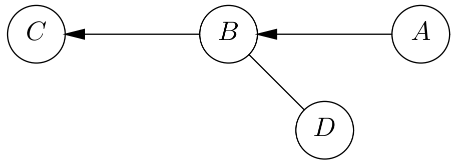
The learning procedure is described in Algorithm 1. It will first identify possible effects of assumables (`Effect`) and to which it has an indirect relation (`Rel`), which can be used to build up a junction tree, where variables are instantiated for a particular value. From this, weights of the assumables can be learned. The properties of junction trees ensure that joint distribution corresponds to the relative frequency of observables, i.e., we have the following, general, result.

Given a chain logic theory T with associated (moralised, triangulated) chain graph G and database D, then after running Algorithm 1, the resulting weight declaration and T will be such that:

$$P_T(O) = \frac{N_O}{N_\varnothing} \tag{5}$$

for all possible observations $O \subseteq \mathcal{O}$.

To illustrate this, suppose we have the following chain graph network:



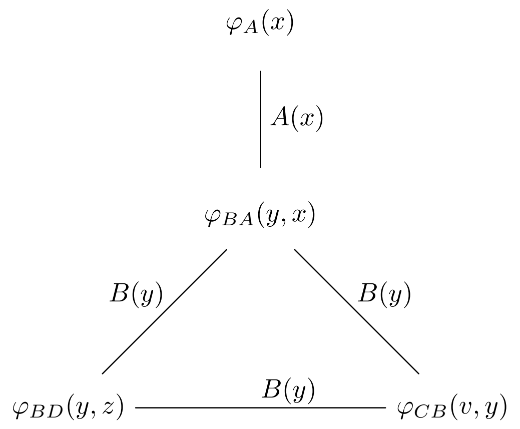
The chain logic program is:

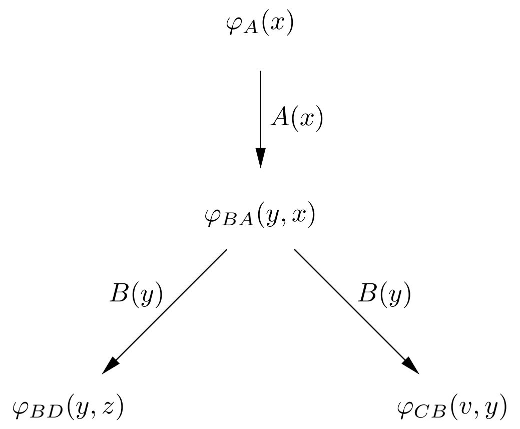
$$\begin{aligned}
&A(x) \leftarrow: \varphi_A(x)\\
&B(x) \leftarrow A(y) : \varphi_{BA}(x, y), \varphi_{BD}(x, z)\\
&C(x) \leftarrow B(y) : \varphi_{CB}(x, y)\\
&D(y) \leftarrow: \varphi_{BD}(x, y)\\
&\perp \leftarrow: \varphi_{BA}(x, y), \varphi_{BD}(\bar{x}, z)
\end{aligned}$$

According to the Algorithm 1, the following graph can be constructed (here shown with quantified variables):

Algorithm 1 learn CL parameters

```
Require: chain logic theory $T$, assumables $\mathcal{A}$, observables $\mathcal{O}$, database $D$
  for $a \in \mathcal{A}$ do
    $\texttt{Effect}(a) \leftarrow \{o \in \mathcal{O} \mid \exists H \in \mathrm{CH} : T \cup H \models o$ and $T \cup (H \setminus \{a\}) \not\models o\}$
    $\texttt{Rel}(a) \leftarrow \{o \in \mathcal{O} \mid \forall H \in \mathrm{CH}: a \in H$ and $T \cup H \models \texttt{Effect}(a)$ implies $T \cup H \models o\}$
  end for
  JG $\leftarrow$ junction graph with nodes $\mathcal{A}$ and separators $\texttt{Rel}(a) \cap \texttt{Rel}(a')$ between adjacent nodes $a$ and $a'$
  $J \leftarrow$ spanning tree of maximal weight of JG, where the weight of an edge is the cardinality of its separator
  DJ $\leftarrow$ any directed tree of $J$
  for $a \in \mathcal{A}$ do
    let $S$ be the union of separators of $a$ with its parents in DJ
    $\texttt{weight}(a) \leftarrow N_{\texttt{Effect}(a) \cup S}/N_S$
  end for
```





A tree can be obtained from this graph by removing any of the edges from the loop in the undirected graph. In fact, it does not matter which one to choose as, if there is a loop, then the separators are all the same set of variables. Equation 5 implies that for chain logic it is irrelevant which one is chosen, but some are more closely related to the original graph as others. For example, if we take $\varphi_A(x)$ as a root, then $\varphi_{BA}(y,x)$ is a conditional probability, as in the original graph. If, on the other hand we take $\varphi_A(x)$ as a leaf, then its weight will be 1 and thus $\varphi_{BA}(y,x)$ will be the *joint* probability of A and B, given its parent.

For obtaining the interpretation of the original graph, choose DJ as the directed tree such that there is an arc from a to a' iff

$$s(a,a') \in \texttt{Effect}(a) \text{ and } s(a,a') \notin \texttt{Effect}(a')$$

where $s(a,a')$ denotes $\texttt{Rel}(a) \cap \texttt{Rel}(a')$, i.e., whenever an assumable a' does not explain an observable it is related to, which means it must be conditioned on this observation. For triangulated chain graphs, such a tree exists. In the example above, we thus take the tree with arrows between φ_A and φ_{BA}, and between φ_{BA} and φ_{CB}, giving, e.g., the following tree:

The learning algorithm will then, e.g., learn that: $\varphi_{CB}(x,y) = N_{\{C(x),B(y)\}}/N_{B(y)}$ which corresponds to exactly the relative frequencies associated to variables in the original chain graph, illustrating the relation between the two formalisms for learning. Relational domains can be represented, and thus learned about using the same machinery, when it is ensured that the underlying graph is characterised by the class of triangulated chain graphs.

References

Ferreira, N., Hommersom, A. J., & Lucas, P. J. F. (2008). From probabilistic Horn logic to chain logic. *Proc. of the BNAIC'08* (pp. 73–80).

Getoor, L., & Taskar, B. (Eds.). (2007). *Introduction to statistical relational learning*. Adaptive Computation and Machine Learning. MIT Press.

Lauritzen, S. L. (1996). *Graphical models*. Oxford:Clarendon.

Poole, D. (1993). Probabilistic Horn abduction and Bayesian networks. *AI Journal*, *64*, 81–129.