quired searching through possible structures and led to less accurate models than learning a BN directly from data.

In experiments on 12 standard datasets, we find that our DN2MN conversion method is orders of magnitude faster than weight learning and often more accurate. For DNs with decision tree conditional distributions, the converted MN was often more accurate than the original DN. With logistic regression conditional distributions, the converted MNs were significantly more accurate than performing weight learning.

Our method has potential applications outside of MN structure learning as well. For domains where data is limited or unavailable, an expert could specify the conditional distributions of a DN, which could then be translated into the joint distribution of an MN. This could be much easier and less error-prone than attempting to specify the entire joint distribution directly.

Our paper is organized as follows. We begin with background on Markov networks and dependency networks in Sections 2 and 3. In Section 4, we describe how to convert consistent dependency networks into Markov networks that represent the exact same probability distribution. In Section 5, we discuss inconsistent conditional probability distributions and methods for improving the approximation quality through averaging. We evaluate our methods empirically in Section 6 and conclude in Section 7.

2 MARKOV NETWORKS

A Markov network (MN) represents a probability distribution over a set of random variables \(X = \{X_1, X_2, \ldots, X_n\}\) as a normalized product of factors:

\[
P(X) = \frac{1}{Z} \prod_i \phi_i(D_i) \tag{1}
\]

\(Z\) is the partition function, a normalization constant to make the distribution sum to one; \(\phi_i\) is the \(i\)th factor, sometimes referred to as a potential function; and \(D_i \subset X\) is the set of variables in the domain of \(\phi_i\).

A probability distribution where \(P(x) > 0\) for all \(x \in X\) is said to be positive. An MN that represents a positive distribution can also be written as a log-linear model. In a log-linear model, the probability distribution is expressed as an exponentiated weighted sum of feature functions \(f_i(D_i)\) rather than as a product of factors:

\[
P(X) = \frac{1}{Z} \exp \left( \sum_i w_i f_i(D_i) \right) \tag{2}
\]

The correspondence between (1) and (2) is easily shown by letting \(f_i = \log \phi_i\) and \(w_i = 1\).

For discrete domains, a common choice is to use conjunctions of variable tests as the features. Each test is of the form \((X_i = v_i)\), where \(X_i\) is a variable in \(X\) and \(v_i\) is a value of that variable. We sometimes abbreviate tests of Boolean variables, so that \((X_i = T)\) is written as \(X_i\) and \((X_i = F)\) is written as \(\neg X_i\). A conjunctive feature equals 1 if its arguments satisfy the conjunction and 0 otherwise.

Any factor represented as a table can be converted into a set of conjunctive features with one feature for each entry in the table. When the factors have repeated values or other types of structure, a feature-based representation is often more compact than a tabular one.

The Markov blanket of a variable \(X_i\), denoted \(MB(X_i)\), is the set of variables that render \(X_i\) independent from all other variables in the domain. In an MN, this set consists of all variables that appear in a factor or feature with \(X_i\). These independencies, and others, are entailed by the factorization in (1).

2.1 INFERENCE

Given an MN, we often wish to answer queries such as the probability of one or more variables given evidence. In general, computing exact marginal and conditional probabilities is \#P-complete [16], so approximate inference algorithms are commonly used instead. One of the most popular is Gibbs sampling [5].

Gibbs sampling is a Markov chain Monte Carlo method that generates a set of random samples and uses them to answer queries. The sampler is initialized to a random state consistent with the evidence. Samples are generated by resampling each non-evidence variable in turn, according to its conditional probability given the current states of the other variables. In early samples, the sampler may be in an unlikely and non-representative state, so these “burn-in” samples are typically excluded from consideration. The probability of a query is estimated as the fraction of samples consistent with the query. For positive distributions, Gibbs sampling will eventually converge to the correct probabilities, but this can take a very long time and convergence can be difficult to detect.

2.2 WEIGHT LEARNING

In maximum likelihood parameter estimation, the goal is to select a set of parameters that maximizes the log-likelihood of the model on the training data. Here we assume that the MN is expressed as a log-linear model with a fixed set of features and that all variables are observed in the training data. Since the log-likelihood of an MN is a concave function of the weights, parameter learning can be framed as a convex optimization problem and solved with standard gradient-based approaches. Since log-likelihood and its gradient are usually intractable to compute exactly and slow to approximate, a commonly-used alternative is pseudo-likelihood. Pseudo-log-likelihood (PLL) [2] is the sum of the conditional log-likelihood of each variable given