Improving Chow-Liu Tree Performance by Mining Association Rules

Kaizhu Huang, Irwin King, Michael R. Lyu, and Haiqin Yang

Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
{kzhuang, king, lyu, hqyang}@cse.cuhk.edu.hk

Abstract. We present a novel approach to construct a kind of tree belief network, in which the "nodes" are subsets of variables of dataset. We call this model Large Node Chow-Liu Tree (LNCLT). This technique uses the concept of the association rule as found in the database literature to guide the construction of the LNCLT. Similar to the Chow-Liu Tree (CLT), the LNCLT is also ideal for density estimation and classification applications. More importantly, our novel model partially solves the disadvantages of the CLT, i.e., the inability to represent non-tree structures, and is shown to be superior to the CLT theoretically. Moreover, based on the MNIST hand-printed digit database, we conduct a series of digit recognition experiments to verify our approach. From the result we find that both the approximation accuracy and the recognition rate on the data are improved with the LNCLT structure, when compared with the CLT.

Key words: Classification, Association Rule, Chow-Liu Tree, Large Node, Bayesian Network.

1 Introduction

One of the interesting problems in Machine Learning is density estimation, i.e., given a training dataset, how can we estimate the data distribution? The estimated distribution can be used to perform classification or prediction.

The Naive Bayesian (NB) network demonstrates good performance in using the estimated distribution to construct classifiers, even when compared with the state-of-the-art classifiers, e.g., C4.5 [27]. With a conditional independency assumption among the features or attributes, i.e., $P(A_i, A_j|C) = P(A_i|C)P(A_j|C)$, with A_i, A_j, $1 \leq i \neq j \leq n$ and C representing the attributes and class variable, respectively, NB estimates the joint probability $P(C, A_1, A_2, \ldots, A_n)$ from data and classifies a specific sample into the class with the largest joint probability. Furthermore, this joint probability can be

J. C. Rajapakse et al. (eds.), *Neural Information Processing: Research and Development*

decomposed into a multiplication form based on its independency assumption. Therefore, the decision function can be written as follows:

$$c = \arg\max_{C_i} P(C_i, A_1, A_2, \ldots, A_n)$$
$$= \arg\max_{C_i} P(C_i) \prod_{j=1}^{n} P(A_j|C_i), \quad (1)$$

where, $P(C_i), P(A_j|C_i)$ are usually estimated empirically.

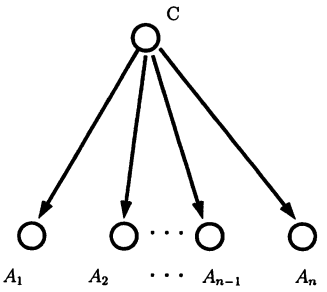


Fig. 1. A Naive Bayesian Classifier. $A_i, 1 \leq i \leq n$, is the attribute. In this figure, the attribute is independent of each other, given the class label C.

The success of the NB is somewhat unexpected since its independency assumption typically does not hold in many cases. A representative example is the so-called "Xor" problem. In this problem, attributes are two binary random variables A and B. When $A = B$, $C = 1$; otherwise $C = 0$. Thus the attribute A is not independent of B, when given the class variable C. NB encounters problems in classifying the "Xor" data. The reason is that $P(C = 0), P(C = 1), P(A = 0|C = 0), P(A = 1|C = 0), P(A = 0|C = 1), P(A = 1|C = 1)$ will all be nearly 0.5 when the data samples are randomly generated. It will be hard to assign any data into the class "0" or "1" since the estimated joint probabilities, according to (1) for both classes, will be about $0.5 \times 0.5 \times 0.5 = 0.125$.

By relaxing the strong assumption, i.e., the independency among the data attributes, of NB, many researchers have developed other types of Bayesian belief networks such as Semi-naive Bayesian networks [18, 12], Selective Naive Bayesian networks [20], and Tree Augmented Naive Bayesian networks [9].

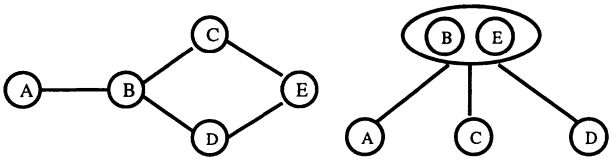
One of the competitive models in this trend is the so-called Chow-Liu Tree (CLT) model [3]. Rather than assuming an independence among the attributes, CLT assumes a tree dependence relationship among the attributes, when given the class variable. The decision function of the CLT constructed from the estimated distribution can be written as a decomposed form:

$$c = \arg\max_{C_i} P(C_i, A_1, A_2, \ldots, A_n)$$
$$= \arg\max_{C_i} P(C_i) \prod_{j=1}^{n} P(A_j | Pa(A_j), C_i), \quad (2)$$

where $Pa(A_j)$ represents the parent node of A_j in the tree structure. The decomposed item $P(A_j|Pa(A_j), C_i)$ is usually estimated empirically.

When compared with NB, CLT can generate a more accurate distribution [3] and achieve lower error rates in classification tasks [9]. Its advantages are partly due to the relaxed restriction than NB [9], its decomposable ability in approximating distribution, and the resistance to over-fitting problems.

However, there are still problems for the CLT, i.e., the tree dependence assumption on the underlying structure of the training dataset will be still too strong to be satisfied in many cases. For a simple example, see Fig. 2(a). If the underlying structure of a dataset can be represented as a graph as Fig. 2(a), the CLT method will not be able to restore this structure, since Fig. 2(a) is not a tree due to its cyclic characteristic.



(a) (b)

Fig. 2. (a): The underlying structure of a dataset (b): A large node tree structure we call "LNCLT"

However, if we combine some nodes as a "large node", then Fig. 2(a) can be represented as a tree. Fig. 2(b) is indeed such a structure, which is compatible with Fig. 2(a), since they both represent "A, C, D are conditionally independent of each other, when given B and E".

Motivated from this finding, we develop a Large Node Chow-Liu Tree (LNCLT), where the large node means a subset of attributes as $\{B, E\}$ in Fig. 2(b). Based on the improved techniques of association rules [1], we propose reasonable combination rules to construct the large node tree structure directly from the draft structure by the CLT. Both theoretical results and experimental results demonstrate the superiority of our novel model over CLT.

One of the important features of our approach is that, implied by its name, the resulting large node tree maintains a tree structure, where the estimated distribution is easily decomposed and therefore naturally enjoys the resistance ability to the overfitting problems.

The main contributions of this paper are described as follows. First, we propose a novel Large Node Chow-Liu Tree, which outperforms Chow-Liu Tree theoretically and experimentally. Second, we develop a theory to determine the threshold used in mining association rules, which is usually set by hand. This will, thus, save the time to adapt the threshold by some intuitive methods such as Cross Validation methods [16].

This paper is organized as follows. In next section, we present the related work. In Sect. 3, we describe the background for this paper including the notations, the CLT algorithm and the concept of the association rule. In Sect. 4, we introduce the main work of this paper, namely, the main theoretical results in guiding the construction of the LNCLT and the practical algorithm. Following that, in Sect. 5, we demonstrate the advantages of the LNCLT based on a serious of experiments. We then conclude this paper with final remarks in Sect. 6. Some of the theoretical and experimental results in Sect. 4 and Sect. 5 have been earlier presented in [11] and are expanded significantly in the current paper, while other sections are new.

2 Related Work

It has been an active research topic to attempt to improve the performance of belief networks by relaxing strong connection assumptions. A number of algorithms are proposed to relax the strong assumption of NB [18, 25, 12, 20, 19, 28, 32]. Similarly, in relaxing the CLT, Malvestuto [22] used acyclic hypergraph and brought out a local heuristic method to search the structure. The similar work to learning hypergraph[1] from data was presented by Srebro et al in [29, 15]. They aimed to solve this problem globally and proposed the approximation method as well. Another school of approaches to extend the CLT is the so-called Bayesian Networks [26]. Instead of assuming a dependence tree structure, this method tried to search the dependence relationship among the attributes from data.

However, the above models suffer from the difficulties in approximating the good distributions from data. As shown by Srebro [29], it is an NP-hard problem to find the optimal hypergraph. Even for the proposed approximation method, it cannot achieve satisfactory result [14]. Furthermore, it is also NP-hard to obtain the optimal Bayesian Networks from data [6]. On the other hand, unrestricted Bayesian Networks do not demonstrate an increase in accuracy even when compared to the simple NB network [9].

Other extensions of the Chow-Liu Tree are also investigated recently. Meila [23] proposed to model distributions as the mixture of the Chow-Liu Trees. Dasgupta and Luby [4] suggested polytree Bayesian networks or trees with oriented edges. Huang et al. invented a discriminative way to training Chow-Liu Trees [13].

[1] Srebro et al. named this structure as Markov network or hypertree.

In this paper, we do not aim to find an optimal large node tree structure. Similar to [23], we perform the upgrading directly on the CLT. Instead of using a linear combination of CLTs, we construct a more relaxed graph structure than the CLT, namely, the large node tree, based on the improved techniques from association rules. Moreover, we theoretically prove that the constructed large node tree has a larger log likelihood than that of the CLT and therefore generate a more accurate distribution approximation.

3 Background

In this section, we first describe the notations used in this paper. Next, the concept of CLT and association rules, rather than the details of these two topics will be introduced.

3.1 Notations

The notation here will largely follow that of [23]. Let V denote a set of n random discrete variables and assume A is a subset of V. We denote x_A as one assignment of the variables in A. Moreover we consider a graph $T = (V, E)$ where V is the vertex set and E is a set of undirected edges. If T is a connected acyclic graph, we call T a tree. If the number of edges $|E|$ in a tree T is equal to the number of vertex minus one: $|V| - 1$, we call T a spanning tree. Let V^* denote a set of the subsets of V, where V^* satisfies the following condition:

$$\cup_{U_i \in V^*} U_i = V, \tag{3}$$

$$U_i \cap U_j = \varnothing \quad with \quad U_i, U_j \in V^*, \quad i \neq j. \tag{4}$$

A large node tree $T^*(V^*, E^*)$ is defined as a tree where V^* is the vertex set satisfying the above conditions and E^* is the set of edges among V^*. Here we can see that each vertex of T^* is actually the subset of V and these subsets have no overlapped variables. Figure 3(b) is an example of a large node tree.

According to the tree decomposition, the distribution encoded in the large node tree can be written into:

$$P^*(x_V) = \frac{\prod_{(u,v)\in E^*} P(x_u, x_v)}{\prod_{v \in V^*} {P^*}_v(x_v)^{deg(v)-1}},$$

where, $deg(v)$ refers to the number of edges which contain v as one vertex. The directed large node tree distribution can be written into:

$$P^*(x_V) = \prod_{v \in V^*} {P^*}_{v|Pa(v)} P^*(x_v | x_{Pa(v)}).$$

The problem of learning Large Node Chow-Liu Tree can be informally stated as: given the training dataset S with s independent observation $x^1, x^2, \ldots, x^s$, find a large node tree structure that match S well, where x^i is the n-dimensional vector, which can be represented as $\{{x_1}^i, {x_2}^i, \ldots, {x_n}^i\}$.

3.2 Chow-Liu Tree

We here introduce the algorithm to construct the CLT from data. We will not talk much about the Chow-Liu Tree techniques. Readers interested in this method can refer to [3].

(1) a) Calculate all the mutual information denoted as $I(X_i, X_j)$, between any two nodes X_i, X_j, where, the mutual information between two variables X, Y is defined as

$$I(X,Y) = \sum_{x,y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}. \tag{5}$$

b) Insert them into a set B.

c) Initiate tree $T(V, E)$ where $V = \{\text{all the nodes of a data set}\}$, $E = \{\}$,

(2) Do until E contains $n - 1$ edges (n is the number of nodes)

a) Find the nodes pair (X_{m_1}, X_{m_2}) with maximum mutual information denoted as Im from B.

b) If no cycle is formed in T when the vertex X_{m_1} is connected with X_{m_2}, add edge (X_{m_1}, X_{m_2}) in E, and delete $Im(X_{m_1}, X_{m_2})$ from B.

c) Otherwise, delete $Im(X_{m_1}, X_{m_2})$ from B

d) Go to (2).

The CLT structure obtained from this algorithm is proved to be the optimal one in the sense of Maximum Likelihood criterion [3].

3.3 Association Rules

Mining association rules is recently under great attentions in data mining [1]. This method can be typically applied in the supermarket database analysis problem. In such a problem, it is interesting to know what other goods customers will buy when they buy a certain type of goods. A representative example is that a large number of customers will buy the butter when they buy the bread. Then *bread* $\rightarrow$ *butter* is called one association rule.

The notation of association rule is that: assuming that $I = \{i_1, i_2, ..., i_n\}$ is a set of items and T is a set of transactions, one transaction is a set of items. We use $X \rightarrow Y$ $(X \bigcap Y = \varnothing)$ associated with a confidence $c \in [0, 1]$ to specify an association rule that means customers will buy X item together with Y item with the confidence level c if a fraction c of the transactions consisting of X also consist of Y. The rule has a *support* s in T, if a fraction s of the transactions in T consist of both X and Y. To make the association reliable, this support s has to be greater than a threshold which is called the minimum support. In our problem, T is the dataset and I is the attributes set. Because we are concerned about the classification accuracy, we fix Y and X as the class variable C and a subset of the attributes, respectively. Since we construct each LNCLT or CLT for each class, mining the association rule will

be reduced to mining all the frequent itemsets X, whose supports are larger than the minimum support.

With regards to the algorithm to mine association rules, we refer the interested readers to [1, 10], since it is out of the scope of this paper to introduce the algorithm in detail. In this paper, we use the algorithm called Apriori developed in [1].

4 Learning Large Node Chow-Liu Tree

In this section, we first define a concept called combination transformation. We then in Sect. 4.1 present the combination rules and give the theoretical justifications why these rules will improve the performance of the draft structure. Following that, we propose the theory on how to determine the minimum support used in the combination rules. Finally, we detail the practical algorithm in Sect. 4.2.

Definition 1 *A combination transformation is defined to be a transformation in a tree structure T. This transformation combines several nodes into a large node and keep the connection relationship of T.*

Figure 3 is an illustration of combination transformation. In Fig. 3, (a) is a tree structure and (b) is the result after a combination transformation. In (b) when nodes D, B are combined, the edge $\overline{BE}$ in (a) will be kept as the edge $\overline{(BD)E}$ in (b).

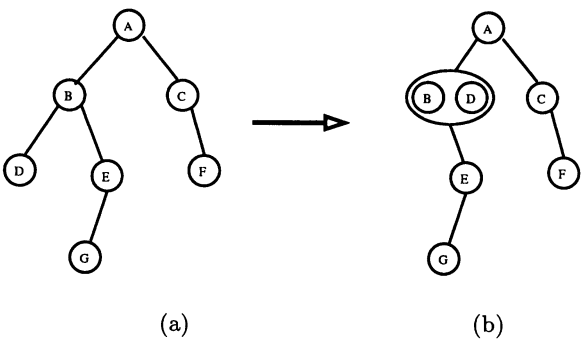


Fig. 3. An illustration of combination transformation

4.1 Main Results

In this subsection, we will present theoretical results on combination rules. We first describe the combination rules.

Rule 1 *Sibling rule: The nodes to be combined satisfy that the set of these nodes are sibling relationship, i.e., there exists another node as their common parent.*

Rule 2 *Parent-child rule: The nodes to be combined satisfy that the set of these nodes can be sorted as a sequence based on a certain node as the root, in which each node is the parent node of its sequent node.*

Rule 3 *Association rule: The nodes to be combined satisfy that, under a given confidence level and a minimum support, the set of these nodes denoted by A forms an association rule , i.e., $A \rightarrow C$, where C is the class label.*

Rule 4 *Bound rule: The nodes to be combined satisfy that the number of these nodes is fewer than a given integer bound K.*

We theoretically show that the resulting graphical structure after a combination transformation satisfying Rule 1 or Rule 2 will have a larger log likelihood and thus can approximate the dataset more accurately. We give Proposition 1, Proposition 2 and further Corollary 1, Corollary 2 to prove this.

We first present a preliminary lemma on the log likelihood of the CLT.

Lemma 1. *Given a training dataset S and n variables defined as in Sect. 3.1, the* log *likelihood $l_t(x^1, x^2, \ldots, x^s)$ of the observations can be written as the following when the dataset is fit as a maximum weight spanning tree, where the weight is given by the mutual information between two nodes:*

$$l_t(x^1, x^2, \ldots, x^s) = \sum_{i=1}^{n} \sum_{k=1}^{s} \log P(x_i^k | \, x_{j(i)}^k), \tag{6}$$

where $j(i)$ represents the parent node of variable i obtained by the ordering based on any certain node as the root in a tree and x^k is an n-dimensional vector $\{x_1^k, x_2^k, \ldots, x_n^k$ with $1 \leq k \leq s\}$. Moreover, this log *likelihood is maximized when the spanning tree is obtained with Chow-Liu Tree method [3].*

The proof can be seen in [3].

Proposition 1. *Given a spanning tree T, if any two nodes satisfy parent-child relationship based on a certain root, then the graphical structure T^* after a combination transformation of these two nodes is, based on the Maximum Likelihood criterion, superior to the original tree T.*

Proof. Using Fig. 4 as an illustration, we assume that the left part (a) is one sub-part of the spanning tree T and in this subpart we perform the combination of two variables X_1 and X_2. To be simple, we assume X_1 has children: X_2, X_q, and X_2 has only one child : X_m. We have the similar proof if X_1 and X_2 have multiple children. Figure. 4(b) is the structure after these two nodes X_1, X_2 with parent-child relationship are combined. For the spanning

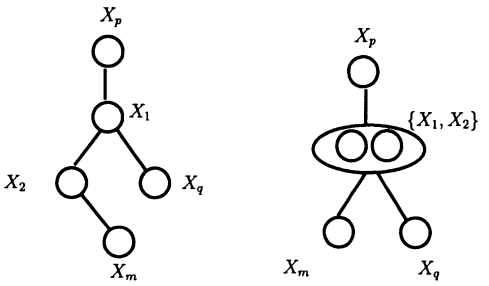


Fig. 4. A parent-child combination (a): The original sub-tree. (b): The result sub-tree after the combination of X_1 and X_2.

tree T, only the subpart (a) is changed into (b) when combining X_1 and X_2 and the other parts of T remain unchanged. We rewrite the log likelihood of the training dataset according to tree T into two parts:

$$\begin{aligned} l_t(x^1, x^2, \ldots, x^s) = & \sum_{i \neq X_1, X_2, X_m, X_q} [\sum_{k=1}^{s} \log P(x_i{}^k \mid x_{j(i)}^k)] + \\ & + \sum_{k=1}^{s} [\log P(x_{X_2}^k \mid x_{X_1}^k) + \log P(x_{X_m}^k \mid x_{X_2}^k) + \\ & + \log P(x_{X_q}^k \mid x_{X_1}^k) + \log P(x_{X_1}^k \mid x_{X_p}^k)]. \end{aligned} \tag{7}$$

The same as (7), we can write the log likelihood encoded in the transformed structure T^* with X_1 and X_2 combined into (8):

$$\begin{aligned} l_{t^*}(x^1, x^2, \ldots, x^s) = & \sum_{i \neq X_1, X_2, X_m, X_q} [\sum_{k=1}^{s} \log P(x_i{}^k \mid x_{j(i)}^k)] + \\ & + \sum_{k=1}^{s} [\log P(x_{X_m}^k \mid x_{X_1}^k x_{X_2}^k) + \log P(x_{X_q}^k \mid x_{X_1}^k x_{X_2}^k) + \\ & + \log P(x_{X_1}^k x_{X_2}^k \mid x_{X_p}^k)]. \end{aligned} \tag{8}$$

Further we can define the second part of (7) as $R(l_t)$ and write it into the entropy form as in (9).

$$
\begin{aligned}
R(l_t) &= \sum_{k=1}^{s}[\log P(x_{X_2}^k \mid x_{X_1}^k) + \log P(x_{X_m}^k \mid x_{X_2}^k) + \\
&+ \log P(x_{X_q}^k \mid x_{X_1}^k) + \log P(x_{X_1}^k \mid x_{X_p}^k)] \\
&= \sum_{k=1}^{s} \log P(x_{X_2}^k \mid x_{X_1}^k) + \sum_{k=1}^{s} \log P(x_{X_m}^k \mid x_{X_2}^k) + \\
&+ \sum_{k=1}^{s} \log P(x_{X_q}^k \mid x_{X_1}^k) - \sum_{k=1}^{s} \log P x_{X_p}^k + \\
&+ \sum_{k=1}^{s} \log P(x_{X_1}^k \mid x_{X_p}^k) + \sum_{k=1}^{s} \log P x_{X_p}^k \\
&= -H(X_2|X_1) - H(X_m|X_2) - H(X_q|X_1) - \\
&- H(X_1X_p) + H(X_p). \qquad (9)
\end{aligned}
$$

In the same way, we can write the second part of (8) into (10).

$$
\begin{aligned}
R(l_{t^*}) &= \sum_{k=1}^{s}[\log P(x_{X_m}^k \mid x_{X_1}^k x_{X_2}^k) + \\
&+ \log P(x_{X_q}^k \mid x_{X_1}^k x_{X_2}^k) + \log P(x_{X_1}^k x_{X_2}^k \mid x_{X_p}^k)] \\
&= -H(X_2|X_1X_p) - H(X_m|X_1X_2) - \\
&- H(X_q|X_1X_2) - H(X_1X_p) + H(X_p). \qquad (10)
\end{aligned}
$$

According to the information theory, we have:

$$
\begin{aligned}
H(X_2|X_1) &\geq -H(X_2|X_1X_p), \\
H(X_m|X_2) &\geq H(X_m|X_1X_2), \\
H(X_q|X_2) &\geq H(X_q|X_1X_2).
\end{aligned}
$$

Therefore, we have the following inequality:

$$
R(l_t) \leq R(l_{t^*}). \qquad (11)
$$

From (7), (8), (11) we obtain that:

$$
l_t \leq l_{t^*}. \qquad (12)
$$

Proposition 1 shows that a single parent-child combination transform will increase the log likelihood of a tree T, which means the data fitness will be increased.

Proposition 2. *Given a spanning tree T, if two nodes satisfy sibling relationship based on a certain root, then the graphical structure T^* after a combination transformation of these two nodes is, based on the Maximum Likelihood criterion, superior to the original tree T.*

The proof of Proposition 2 is much similar to Proposition 1, we will not prove it here.

Based on a sequence of combination transformation, We can easily expand Proposition 1 and Proposition 2 into the following Corollary 1 and Corollary 2.

Corollary 1. *Given a spanning tree T, if a subset of nodes can be sorted as a sequence based on a certain node as the root, in which each node is the father of its sequent node, then the graphical structure T_* after a combination transformation of these nodes in this subset is, based on the Maximum Likelihood criterion, superior to the original tree T.*

Corollary 2. *Given a spanning tree T, if all the nodes in a subset are sibling relationship, then the graphical structure T_* after a combination transformation of all the nodes in this subset is, based on the Maximum Likelihood criterion, superior to the original tree T.*

These two corollaries prove that the combination transformation of parent-child relationship and siblings relationship will increase the approximation accuracy on data. Another advantage of combining the nodes with parent-child or sibling relationship lies in that the transformed graphical structure will maintain a tree structure, which is easily decomposed and enjoys the ability to resist the overfitting problem. On the other hand, combining nodes without parent-child or sibling relationship may result in a non-tree structure. Such example can be seen in Fig. 5.

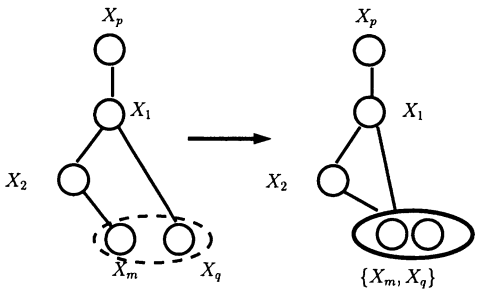


Fig. 5. An example to illustrate that combining nodes without parent-child or sibling relationship may result in a non-tree structure.

Here we argue that Rule 3 is reasonable. Since those attributes with an association rule pointing to class label C will occur with one another more frequently, they should be more dependent on one another than other attributes. Thus they are more like a single node and should be combined with the higher priority.

On the other hand, Rule 4 is also necessary. The bound K cannot be too large, or the estimation of the probability of the large node will be unreliable. An extreme case is that, when K is equal to n, i.e., the number of the

attributes, all the nodes will be combined into one large node. In this case the estimated distribution will be the empirical distribution, which is very unreliable to represent the data.

Until now, we do not mention how to set the threshold, i.e., the minimum support in using the associate rules. In the next section, we present how to determine the minimum support theoretically.

4.2 How to Determine the Minimum Support?

Without loss of generality, we begin with the $2-1$ association rule:$X \rightarrow Y$, which means X contains just two attributes $X = \{i, j\}$ and Y contains one variable $Y = \{l\}$ (in our problem, l is the class variable). The derivation for the general case will be similar. In the following, we use the Chebyshev Theorem to derive the suitable minimum support.

This theorem gives the lower bound on the probability that the frequency f of an event c after n trials differs from the real probability p within a ε variation:

$$P(|f - p| \leq \varepsilon) \geq 1 - \frac{p(1-p)}{\varepsilon^2 n}. \tag{13}$$

In our problem, the frequency is given by

$$f = \frac{N_{ijl}}{N_{ij}},$$

where, N_{ij} is defined as the number of the occurrence of the item $\{i, j\}$ and N_{ijl} is similarly defined. The value p is defined as the real probability of the event "the itemsets which consist of i, j will also consist of l". If we rewrite the absolute form of (13),we can have the following:

$$f - \varepsilon \leq p \leq f + \varepsilon.$$

In the association rule mining process, it is required that p is greater than the confidence level p_{cf} and p is also has to be less than 1. So we can simply specify that:

$$f - \varepsilon = p_{cf},$$
$$f + \varepsilon = 1.0.$$

From above, we obtain: $\varepsilon = \frac{(1-p_{cf})}{2}$. Combined this with (13), we have the following:

$$
\begin{aligned}
P(|f-p| \leq \varepsilon) &\geq 1 - \frac{p(1-p)}{\varepsilon^2 n} \\
&= 1 - \frac{p(1-p)}{\frac{(1-p_{cf})^2}{2} n} \\
&\geq 1 - \frac{0.5(1-0.5)}{\frac{(1-p_{cf})^2}{2} n} \\
&= 1 - \frac{1}{(1-p_{cf})^2 n}. \qquad (14)
\end{aligned}
$$

In order to obtain reliable association rule, the frequency: $f = \frac{N_{ijl}}{N_{ij}}$ has to be close to the real probability of c event. So the probability that the frequency is close to the real probability must be at least greater than 0.5, which implies:

$$
1 - \frac{1}{(1-p_{cf})^2 n} \geq 0.5. \qquad (15)
$$

Here n is equal to N_{ij}, which at least achieves a number

$$
n = N_{ij} \geq s_m N, \qquad (16)
$$

where N is the number of the cases or samples in dataset, and s_m is the minimum support. To satisfy (15), n should be big enough. Thus its lower bound $s_m N$ should be big enough. At last we obtain the bound of the minimum support:

$$
s_m \geq \frac{2}{(1-p_{cf})^2 N}.
$$

In a word, the above can be written into a lemma:

Lemma 2. *In order to make the inference in mining association rule reliable, the minimum support of the association rule must satisfy the following inequality:*

$$
s_m \geq \frac{2}{(1-p_{cf})^2 N}, \qquad (17)
$$

where N is the total number of cases in dataset, p_{cf} is the confidence level specified by the user.

4.3 Practical Algorithm

In this section, we describe the detailed algorithm to build up Large Node Chow-Liu Tree from data. Our algorithm consists of three phases. In the first phase we utilize Apriori in [1] to detect all the association rules satisfying

Rule 4. The second phase is basically the Chow-Liu Tree construction algorithm. In the last phase, we combine the attributes, which satisfy combination rules and have higher supports, and upgrade the Chow-Liu Tree structure into the LNCLT structure iteratively.

Phase 1: Detecting all the association rules $X \rightarrow Y$, where Y is specified by the class variable C and X is a subset of attributes set, with the cardinality fewer than a bound K.

(1) Determine a good value of the minimum support, based on (17). Call the Apriori procedure to generate the association rules, whose X's have the cardinality fewer than K.
(2) Record all the association rules together with their supports into list L.

Phase 2(3): Drafting Chow-Liu Tree [3].

Phase 3: Adapting the tree structure based on combination transformation

(4) According to tree T, filter out association rules from L whose X's do not satisfy combination conditions, i.e., Rule 1 or Rule 2 from L. We get the new $L^{'}$.
(5) Sort $L^{'}$ in descending order based on the supports of the association rules.
(6) Do until $L^{'}$ is $NULL$.
    (a) Do the combination transformation based on the first itemset l_1 of $L^{'}$.
    (b) Delete l_1 and any other association rules l_i in $L^{'}$ which satisfy the following condition:

$$l_1.X \cap l_i.X \neq \varnothing,$$

    where $l_1.X$ and $l_i.X$ refers to the X part of l_1 and l_i, respectively.
    (c) Examine whether the newly generated items satisfy the combination rules. If yes, insert them into $L^{'}$ and sort $L^{'}$.
    (d) Go to (a).

5 Experiments

In this section, we first present the setup information of our experiments. Following that, we describe our pre-processing methods including handling zero-counts problems and feature extraction. In Sect. 5.3, we demonstrate the experimental results.

5.1 Setup

Our experiments are implemented on MNIST datasets [21]. The MNIST datasets consist of a 60000-digit training dataset and a 10000-digit test

dataset. Both the training dataset and the test dataset consist of 28×28 gray-level pixels digits. As mentioned before, the bound K in Rule 4 cannot be set to a big value, we set K to 3 in our experiment.

5.2 Pre-Processing Methods

5.2.1 Feature Extraction Methods

We use the same method as [2] to extract 96-dimensional binary features from the digits. Since this method requires the binarization of the images, we first use a global threshold to binarize the training and test dataset. Then we segment the digit images into 2×3 sub-regions uniformly. In each sub-region, we judge whether four configurations given in Fig. 6 and their rotated configurations in other three main directions exist. Each configuration corresponds to a binary feature; therefore, the total number of the features will thus be $2 \times 3 \times 4 \times 4 = 96$.

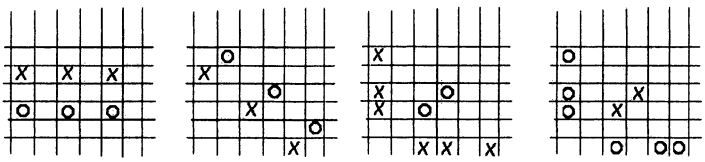


Fig. 6. Four configurations to extract features with $\times$'s and $\circ$'s representing black pixels and white pixels respectively. These configurations will be rotated clockwise with angles $90^o, 180^o$ and 270^o, respectively

5.2.2 Attacking Zero-Counts Problem

Zero-counts problems happen when a given class label and some value of the attribute never occur in the training dataset. This may cause problems in using the estimated probabilities to construct the decision function. For example, in the CLT's decision function, if one value a_j^k for an attribute A_j is never achieved, the empirically estimated $P(A_j = a_j^k | A_l = a_l^p, C = C_i)$ will be zero. Consequently, when $A_j = a_j^k$, the joint probability in the right part of (2) will be 0, whatever its parent is and the other terms are. Similar problems also happen for LNCLT. To tackle this problem, we use the popular Laplace correction method [24]. The modified estimated empirical probability for $P(A_j = a_j^k | A_l = a_l^p, C = C_i)$ is given by

$$(\#(A_j = a_j^k, A_l = a_l^p, C = C_i) + f)/(\#(A_l = a_l^p, C = C_i) + fm_j), \quad (18)$$

instead of the uncorrected one,

$$\#(A_j = a_j^k, A_l = a_l^p, C = C_i)/\#(A_l = a_l^p, C = C_i), \tag{19}$$

where m_j is the number of values for attribute A_j, $\#(A_j = a_j^k, C = C_i)$ denotes the number of the occurrence that the attribute A_j achieves its k-th value a_j^k and the class label C achieves C_i. Other $\#(\cdot)$'s are similarly defined. We take the same value $1/N$ for parameter f as [5, 17], where N is the number of samples in training database. The technique is similarly used in estimated the probability of large nodes in LNCLT.

5.3 Results

In this subsection, we compare the performance of the LNCLT with the CLT in the tasks of approximating the dataset and performing classification. We built 10 LNCLTs for 10 digits. When examining the performance in approximating the dataset of the LNCLT and the CLT, we use the log likelihood criterion. When performing classification, we calculate the 10 probabilities for the test sample based on 10 LNCLTs and output the digit, whose LNCLT has the maximum probability, as the result.

5.3.1 Log Likelihood

Table 1. Minus Log Likelihood

Digit	Training (bits/digit)		Testing (bits/digit)	
	LNCLT	CLT	LNCLT	CLT
0	30.14	30.87	30.05	31.00
1	13.08	13.75	12.12	12.86
2	33.78	34.68	33.03	34.05
3	34.49	35.51	33.87	34.95
4	27.98	28.70	27.58	28.34
5	32.45	33.17	32.31	33.18
6	26.96	27.63	26.60	27.26
7	25.01	25.83	24.84	25.79
8	34.15	34.94	33.75	34.58
9	26.90	27.52	26.12	26.63

From Table 1, we can see that the log likelihood of the LNCLT is larger than that of the CLT for all the ten digits both in training dataset and test dataset. This result shows that the LNCLT approximates the data more accurately, which is consistent with our theoretical analysis in the previous sections.

5.3.2 Recognition Rate

We first use the 60000-digit training dataset to train the LNCLT and CLT. To test the performance of the LNCLT and CLT, we extract 1,000 digits from the 10000-digit test dataset randomly as our test dataset. We do the 1000-digit test for 10 times to evaluate the performance difference between the LNCLT and CLT. Table 2 describes the result. From Table 2, it is clearly observed that the LNCLT performs better than CLT in all of 10 test datasets. We note that, when compared with the results of other approaches on MNIST, the recognition rate here is relatively low. The simple binarization method and different feature extraction method may partly explain this phenomenon.

Table 2. Recognition Rate

Dataset	1	2	3	4	5
CLT(%)	83.20	84.70	84.10	83.50	83.70
LNCLT(%)	83.70	85.90	84.70	84.20	84.90
Dataset	6	7	8	9	10
CLT(%)	85.10	84.30	83.30	83.50	83.80
LNCLT(%)	86.00	85.40	83.50	83.90	85.70

6 Conclusion

In this paper, we have described a method for constructing a kind of "tree" belief network: Large Node Chow-Liu Tree. This method can be seen as the extension of Chow-liu Tree algorithm. With the combination of improved association rule techniques, our novel model can partially overcome the disadvantages of Chow-Liu Tree, i.e., the inability to represent non-tree structures and maintain the advantages of Chow-Liu Tree, such as the decomposition ability in estimating the distribution. we demonstrate that the Large Node Chow-Liu Tree is superior to the CLT both theoretically and experimentally.

Two issues need to be checked in the near future. First, although the LNCLT model achieves performance superior to the CLT model, the proposed iterative process of combining nodes into large nodes may be time-consuming. How to reduce the time-complexity thus becomes a part of our future work. Second, the parameter K, namely, the maximum number of nodes which can be combined, is simply set to 3, which is unnecessarily the optimal value. How To investigate parameter selection methods such as [7, 8, 30, 31] and propose efficient algorithms remains one of our research directions.

Acknowledgment

This research is supported fully by grants from the Hong Kong's Research Grants Council (RGC) under CUHK 4407/99E and CUHK 4222/01E.

References

1. R. Agrawal and R. Srikant. Fast algorithms for mining association rules. *Proceedings of International Conference on Very Large Data Bases (VLDB-1994)*, 1994.
2. R. Bakis, M. Herbst, and G. Nagy. An experimental study of machine recognition of hand-printed numerals. *IEEE Transactions on Systems Science and Cybernetics*, SSC-4(2), JULY 1968.
3. C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Trans. on Information Theory*, 14:462–467, 1968.
4. S. Dasguota. Learning polytrees. In *Uncertainty in Artificial Intelligence*, 1999.
5. P. Domingos and M. J. Pazzani. On the optimality of the simple baysian classifier under zero-one loss. *Machine Learning*, 29:103–130, 1997.
6. J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In *International Conference on Machine Learning*, pages 194–202, 1995.
7. G. Elidan, N. Lotner, N. Friedman, and D. Koller. Discovering hidden variables:a structure-based approach. In *NIPS 13*, 2001.
8. N. Friedman and G. Elidan. Learning the dimensionality of hidden variables. In *Proceedings of Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI)*, 2001.
9. N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–161, 1997.
10. J. Hipp, U. Guntzer, and G. Nakhaeizadeh. Algorithms for association rule mining-a general survey and comparison. *ACM SIGKDD Explorations*, 2:58–64, July 2000.
11. K. Huang, I. King, and M. R. Lyu. Constructing a large node chow-liu tree based on frequent itemsets. In Lipo Wang, Jagath C. Rajapakse, Kunihiko Fukushima, Soo-Young Lee, and Xi Yao, editors, *Proceedings of the International Conference on Neural Information Processing (ICONIP-2002), Orchid Country Club, Singapore*, pages 498–502, 2002.
12. K. Huang, I. King, and M. R. Lyu. Learning maximum likelihood semi-naive bayesian network classifier. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics (SMC-2002), Hammamet, Tunisia*, 2002.
13. K. Huang, I. King, and M. R. Lyu. Discriminative training of bayesian chow-liu tree multinet classifiers. In *Proceedings of International Joint Conference on Neural Network(IJCNN-2003), Oregon, Portland, U.S.A.*, volume 1, pages 484–488, 2003.
14. K. Huang, I. King, and M. R. Lyu. Finite mixture model of bound semi-naive bayesian network classifier. In *Joint 13th International Conference on Artificial Neural Network (ICANN-2003) and 10th International Conference on Neural Information Processing (ICONIP-2003), Long paper, Lecture Notes in Computer Science*, pages 115–122, 2003.
15. D. Karger and N. Srebro. Learning markov networks: maximum bounded tree-width graphs. In *Symposium on Discrete Algorithms*, pages 392–401, 2001.
16. R. Kohavi. A study of cross validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-1995)*, pages 338–345. San Francisco, CA:Morgan Kaufmann, 1995.

17. R. Kohavi, B. Becker, and D Sommerfield. Improving simple bayes. In *Technique report.* Data Mining and Visualization Group, Silicon Graphics Inc., Mountain View, CA., 1997.
18. I. Kononenko. Semi-naive bayesian classifier. In *Proceedings of Sixth European Working Session on Learning*, pages 206–219. Springer-Verlag, 1991.
19. P. Langley. Induction of recursive bayesian classifiers. *In Proceedings of the 1993 European Conference on Machine learning*, pages 153–164, 1993.
20. P. Langley and S. Sage. Induction of selective bayesian classifiers. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence (UAI-1994)*, pages 399–406. San Francisco, CA: Morgan Kaufmann, 1994.
21. Y. Le Cun. http://www.research.att.com/ yann/exdb/mnist/index.html.
22. F. M. Malvestuto. Approximating discrete probability distributions with decomposable models. *IEEE Transactions on Systems, Man and Cybernetics*, 21(5):1287–1294, 1991.
23. M. Meila and M. Jordan. Learning with mixtures of trees. *Journal of Machine Learning Research*, 1:1–48, 2000.
24. T. Niblett. Constructing decision trees in noisy domains. In *Proceedings of the Second European Working Session on Learning*, pages 67–78, 1987.
25. M. J. Pazzani. Searching dependency in bayesian classifiers. In D. Fisher and H.-J. Lenz, editors, *Learning from data: Artificial intelligence and statistics V*, pages 239–248. New York, NY:Springer-Verlag, 1996.
26. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: networks of plausible inference.* Morgan Kaufmann, CA, 2nd edition, 1997.
27. J. R. Quinlan. *C4.5 : programs for machine learning.* San Mateo, California:Morgan Kaufmann Publishers, 1993.
28. M. Sahami. Learning limited dependence bayesian classifiers. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 335–338. Portland, OR:AAAI Press, 1996.
29. N. Srebro. Maximum likelihood bounded tree-width markov networks. *MIT Master thesis*, 2001.
30. A. Stolcke and S. Omohundro. Hidden markov model induction by bayesian model merging. In *NIPS 5*, pages 11–18, 1993.
31. A. Stolcke and S. Omohundro. Inducing probabilistic grammars by bayesian model merging. In *International Conference on Grammatical Inference*, 1994.
32. G. Webb and M. J. Pazzani. Adjusted probability naive bayesian induction. In *the eleventh Australian Joint Conference on Artificial Intelligence*, 1998.