A tree augmented classifier based on Extreme Imprecise Dirichlet Model

Giorgio Corani
IDSIA
Manno, Switzerland
giorgio@idsia.ch

Cassio P. De Campos
IDSIA
Manno, Switzerland
cassio@idsia.ch

Sun Yi
IDSIA
Manno, Switzerland
yi@idsia.ch

Abstract

In this paper we present TANC, i.e., a tree-augmented naive credal classifier based on imprecise probabilities; it models prior near-ignorance via the Extreme Imprecise Dirichlet Model (EDM) [1] and deals conservatively with missing data in the training set, without assuming them to be missing-at-random. The EDM is an approximation of the global Imprecise Dirichlet Model (IDM), which considerably simplifies the computation of upper and lower probabilities; yet, having been only recently introduced, the quality of the provided approximation needs still to be verified. As first contribution, we extensively compare the output of the naive credal classifier (one of the few cases in which the global IDM can be exactly implemented) when learned with the EDM and the global IDM; the output of the classifier appears to be identical in the vast majority of cases, thus supporting the adoption of the EDM in real classification problems. Then, by experiments we show that TANC is more reliable than the precise TAN (learned with uniform prior), and also that it provides better performance compared to a previous [13] TAN model based on imprecise probabilities. TANC treats missing data by considering all possible completions of the training set, but avoiding an exponential increase of the computational times; eventually, we present some preliminary results with missing data.

Keywords. Imprecise Dirichlet Model, Extreme Imprecise Dirichlet Model, Classification, TANC, Naive Credal Classifier.

1 Introduction

Classifiers based on imprecise probabilities are progressively becoming known and appreciated also outside the area of imprecise probabilities [2]; typically, they are based on the Imprecise Dirichlet Model (IDM) to model a condition of prior *near-ignorance*. When faced with an instance whose classification is prior-dependent, they preserve reliability by returning a set of classes (*indeterminate classifications*) instead of a single class. Thanks to the IDM, credal classifiers robustly deal with cases where the evidence arising from the data is not strong enough to smooth the effect of the prior choice.

Two IDM variants have been adopted in credal classifiers: the global IDM or the local IDM; the local lacks some constraints present in the global. The global IDM can make it very difficult to solve the optimization problem to determine lower and upper probabilities. So far, the naive credal classifier (NCC) of [10] is the only case in which it has been possible to develop a credal classifier based on the global IDM. On the contrary, the local IDM allows for an easier solution of the optimization problem; yet, it can return probability intervals that can be unnecessarily wide, compared to the global IDM.

Recently, the EDM (Extreme Dirichlet Model) [1] has been introduced; it restricts the credal set of the global IDM only to its extreme distributions. The intervals returned by the EDM are hence included in the intervals returned by the global IDM; however, the EDM can considerably simplify the solution of the optimization problem. So far, the EDM has been used only in very preliminary experiments; as recognized also in [1], it is still necessary to test the EDM in real classification problems and to study the difference with the global IDM. A first contribution of this paper is that we have implemented NCC with EDM and we have compared it (using 23 data sets) against NCC with global IDM; results show that the two models returns the same set of classes in the large majority of cases.

However, besides prior-ignorance, there is another kind of ignorance involved in the process of learning from data, i.e., ignorance about the missingness process. Usually, classifiers ignore missing data; this entails the idea that the missingness process (MP) is non-selective in producing missing data, i.e., it is MAR (*missing at random* [6]). However, assuming MAR cannot be regarded as an objective-minded approach, if one is ignorant about the MP. According to the Conservative Updating Rule [11, 12], in order to deal conservatively with nonMAR[1]

[1] The term nonMAR is used to indicate that MAR is not assumed.

missing data in the training set, it is necessary to compute a likelihood for each possible completion of the data set. The naive credal classifier of [10] implements such an approach for data that are missing in the training set.

However, naive classifiers can become inadequate on certain data sets, as they assume the statistical independence of the features given the class. Tree augmented naive classifiers [5] have been shown to often outperform naive Bayes, as they can model more realistically complex data sets. An attempt to extend TAN to imprecise probabilities has been proposed in [13]; in the following, this algorithm is referred to as TANC*. TANC* is based on the local IDM, to keep the computation affordable; yet, this choice is likely to make TANC* much more indeterminate than if the global IDM was used. In fact, TANC* returns a considerable number of indeterminate classifications [13]. A further characteristic of TANC* is that it assumes missing data to be MAR, which also contributes for its efficiency.

In this paper we present TANC, i.e., a tree-augmented naive credal classifier based on imprecise probabilities, which (a) models prior near-ignorance via the EDM and (b) treats missing data in the training set[2] without assuming MAR, thus computing a set of likelihoods. Although the number of possible likelihoods is in principle exponential with respect to the number of missing values, we show that the computational complexity of TANC does not necessarily increase exponentially with the total number of missing data in the training set.

We thoroughly evaluate TANC by experiments. Firstly, we evaluate TANC against the precise TAN (i.e., learned with uniform prior) on several data sets; we show that TANC is effective at detecting hard-to-classify instances, over which TAN becomes unreliable; instead, TANC preserve its reliability thanks to indeterminate classifications. In a second series of experiments, we compare TANC and TANC*; we show that TANC is less indeterminate than TANC*; the results suggest moreover that TANC returns determinate and correct answers on instances over which TANC* is unnecessarily indeterminate. Since the difference between TAN, TANC and TANC* lies in the model of prior ignorance, the differences between them decreases with the size of the data set: large amount of data reduce the role of prior densities.

Eventually, we present some preliminary results with non-MAR missing data, comparing TANC against the naive credal classifier (which is also able to treat missing data as nonMAR). Under this setting, TANC appears to be much more indeterminate than the naive credal classifier, because of the more complex graph.

The paper is structured as follows: Section 2 introduces the notation and the basic definitions; Section 3 describes the Imprecise Dirichlet Model in its local, global and extreme specifications; in Section 5 we experimentally show that using the naive credal classifier with global IDM or with the EDM leads to equivalent classifications in most cases. Section 6 presents the TANC algorithm and proves its correctness; Section 7 shows the experimental results, including the comparison against TAN, TANC* and some preliminary results with missing data. Finally, Section 8 contains the conclusions.

[2]The extension to nonMAR missing data in the testing set is left for future development.

2 Notation and Basic Definitions

This section presents the notation used later in the paper, the definition of a credal network and the specification of the data that is employed for learning the parameters of the network. To simplify, we use a definition of credal network were the factorization is enforced in a set of joint probability distributions.

Definition 1 *A credal network is a triple $(\mathcal{G}, \mathcal{X}, \mathcal{K})$, where $\mathcal{G}$ is a directed acyclic graph with nodes associated to discrete random variables $\mathcal{X} = \{X_1, \ldots, X_m\}$ and $\mathcal{K}$ is a set of multinomial probability distributions on $\mathcal{X}$ such that each $p \in \mathcal{K}$ factorizes as $p(\mathcal{X}) = \prod_i p(X_i|\Pi_i)$ (which can be read as* every variable is conditionally independent of its non-descendants given its parents*), where Π_i denotes the parents of X_i in $\mathcal{G}$ (when $\Pi_i = \emptyset$, $p(X_i|\Pi_i)$ is in fact the marginal $p(X_i)$).*

The state space of a variable X_i is denoted by Ω_{X_i}, and the joint space on a set of variables $\mathcal{Y}$ by $\Omega_{\mathcal{Y}} = \times_{X \in \mathcal{Y}} \Omega_X$. Lowercase letters are used to specify assignments to variables: $x_i \in \Omega_{X_i}$ is a category of X_i, and $\pi_i \in \Omega_{\Pi_i}$ is an assignment to all parents of X_i. Parents and children of variables are denoted always with respect to the graph $\mathcal{G}$ of the network. A variable X_i with $\Pi_i = \emptyset$ is called a *root* variable. We further denote by Λ_i the set of children of X_i.

We assume the training data set D to contain n instances of type $\mathbf{x} = \{x_1, \ldots, x_m\}$. With reference to the subset of variables $\mathcal{Y} \subseteq \mathcal{X}$, we define $n_{\mathbf{y}}$ as the number of instances for which the set of variables $\mathcal{Y}$ is set to $\mathbf{y}$.

We allow the training data set to contain missing values, that is, for each instance $\mathbf{x}$ some of its elements may be absent. A *completion* of $\mathbf{x}$ is an assignment to the missing values such that $\mathbf{x}$ becomes complete. A completion of the data set is a completion for all its instances. We denote by $\mathbf{d}_{\mathcal{Y}}$ a possible realization of the training data set (i.e., the observed values plus a possible realization for missing data, if any) restricted to the variables $\mathcal{Y} \subseteq \mathcal{X}$.

3 Variants of the Imprecise Dirichlet Model

The *Imprecise Dirichlet Model*(IDM) [8] is a tool for inference from categorical data, based on a *set* of prior Dirichlet densities. In the following, we illustrate the different variants of the IDM, considering as an example the simple credal network $X_1 \rightarrow X_2$.

As for the marginal distribution $p(X_1)$, the Dirichlet density is proportional to $\prod_{x_1 \in \Omega_{X_1}} \theta_{x_1}^{\alpha_{x_1}-1}$, where $\alpha_{x_1} > 0$ and $\sum_{x_1 \in \Omega_{X_1}} \alpha_{x_1} = s$, where s represents the *equivalent sample size* (or *hidden instances*), which determines the weight of the prior compared to the total number of instances in the training set. By letting the hyper-parameters α_{x_1} take all the possible values in their domain of definition, the IDM produces an interval posterior estimate of the chance, which for each $x_1 \in \Omega_{X_1}$ is:

$$\left[\frac{n_{x_1}}{n+s}, \frac{n_{x_1}+s}{n+s}\right]. \quad (1)$$

When we move to the estimation of $p(x_2|x_1)$, the IDM can be applied locally or globally. By the *local* IDM, we repeat the estimation of formula (1), thus obtaining:

$$\left[\frac{n_{x_1x_2}}{n_{x_1}+s}, \frac{n_{x_1x_2}+s}{n_{x_1}+s}\right]. \quad (2)$$

In other terms, the hyper-parameters $\alpha_{x_1x_2}$ can vary between 0 and s ($0 < \alpha_{x_1x_2} < s$). In this way, we obtain a local credal set for each variable and each assignment of its parents; the global credal set is eventually obtained by the multiplication of the local credal sets.

Alternatively, one can use the *global* IDM; in this case the hyper-parameter $\alpha_{x_1x_2}$ is constrained by $\sum_{x_2} \alpha_{x_1x_2} = \alpha_{x_1}$, where α_{x_1} is the hyper-parameter of the marginal distribution of the parent. The intervals computed by the global IDM are:

$$\left[\frac{n_{x_1x_2}}{n_{x_1}+\alpha_{x_1}}, \frac{n_{x_1x_2}+\alpha_{x_1}}{n_{x_1}+\alpha_{x_1}}\right]. \quad (3)$$

The global IDM estimates narrower posterior intervals than the local IDM because of these additional constraints. In fact, the intervals computed by the local IDM can be very wide when we analyze the corresponding set of joint distributions. On the other hand, under the global IDM, it is usually hard to solve inferences, because the parameters of the network become all correlated in some way. One of the few cases in which this computation is tractable is the naive credal classifier [10].

The EDM is a modification of the global IDM which restricts the IDM to its extreme distributions. Let us consider X_1 again; the EDM allows α_{x_1} to assume two values: 0 or s; hence, it does *not* consider all the Dirichlet distributions defined by the constraint $\sum_{x_1 \in \Omega_{X_1}} \alpha_{x_1} = s$, which are infinite. Analogously, $\alpha_{x_1x_2}$ can assume only two values: 0 or s, but still depends on α_{x_1}. In fact, the EDM treats the s hidden instances as s rows of missing data; the rows are assumed to be identical, but there is ignorance about the value assumed by each variable; such an ignorance determines the credal set.

When applied to a single variable, EDM returns the same interval of the global IDM; however, when applied to a credal network, it returns intervals that are included (or at most equivalent) in the intervals computed by the global IDM [1].

4 Credal Classification

We denote the class variable as C, assuming values in Ω_C; while the set of remaining variables $\mathcal{Y} = \mathcal{X} \setminus C$ are called *features*. The goal of classification is to build a classifier on a training set, and then to predict the unknown class of new instances, given the values $\mathbf{y}$ of the features.

According to [7], the optimality criterion for classification based on imprecise probabilities is to return the *non-dominated* classes. In particular, given the values $\mathbf{y}$ of the features, class *c'* dominates (or *credal-dominates*) class *c"* if and only if:

$$\min_{p \in \mathcal{K}} (p(c'|\mathbf{y}) - p(c''|\mathbf{y})) > 0$$

The set of non-dominated classes can be detected by performing repeated pairwise comparisons, as shown in Figure 1.

IDENTIFICATION OF NON-DOMINATED CLASSES

1. set NonDominatedClasses := Ω_C;
2. for class $c' \in \Omega_C$
   - for class $c'' \in \Omega_C$, $c'' \neq c'$
     - if c'' is dominated by c', drop c'' from NonDominatedClasses and break the internal loop;
3. return NonDominatedClasses.

Figure 1: Identification of non-dominated classes via pairwise comparisons.

A key point is that there can be several non-dominated classes and that these classes are incomparable; in this case, the classifier returns an indeterminate (or set-valued) classification. Classifiers that issue set-valued classifications are called *credal classifiers* . Intuitively, credal classifiers will return determinate classifications (i.e., a single class) on easy-to-classify instances, and more classes on hard-to-classify instances.

5 IDM vs. EDM: empirical comparison on Naive Credal Classifier

Before describing TANC, we experimentally evaluate the naive credal classifier with adoption of the EDM (NCC-EDM) against the traditional naive credal classifier based on the global IDM (NCC). When checking credal-dominance between c' and c'', NCC searches the minimum of $p(c')/p(c'')$ over $(0, s)$, while NCC-EDM evaluates the ratio $p(c')/p(c'')$ only in 0 and s. We have implemented NCC with EDM by reworking the code of JNCC2 [3], an open source implementation of NCC.

The answers returned by NCC and NCC-EDM might be different: when checking whether c' credal-dominates c'', it can happen that NCC-EDM detects credal-dominance while NCC, using a larger credal set (which by the way contains the former), does not detect credal-dominance (in other terms: NCC can find a lower minimum, implying non-dominance, than NCC-EDM).

Some of this different dominance tests do not affect the final set of non-dominated classes, because several pairwise comparisons are run, but some do. Therefore, NCC and NCC-EDM may return distinct sets of non-dominated classes.

To empirically evaluate the difference between NCC and NCC-EDM we have worked on 23 data sets from the UCI repository [4]. Each data set has been used as training and then as testing set; in fact, the goal here is to compare the answers of the two classifiers and not to provide an assessment of their accuracy.

On 22 data sets out of 23, the percentage of credal-dominance tests which receive a different answer from NCC-EDM and NCC is far smaller than 1%; the percentage of instances over which the two models return a different set of dominated classes is very low: 0.01% on average. The number of performed pairwise comparison overall is in the order of 10^6, while the total number of instances classified by NCC and NCC-EDM is around 10^5.

There is however a single data set over which NCC and NCC-EDM lead to different results: the *audiology*. It has 226 instances, 24 classes and 69 features. Remarkably, most binary features have *very* skewed distributions, such as 224 versus 2, or 225 versus 1. Because of the many classes and of the unevenly distributed features, the differences on the model of prior ignorance can lead to a different set of non-dominated classes. This happens on 51/226 instances, i.e., about 22% of the instances.

We conclude that NCC and NCC-EDM are practically equivalent on most cases; however, differences between the two models can arise on data sets with many classes and unevenly distributed features. Still, such indications support the introduction of EDM in classification.

[3] http://www.idsia.ch/~giorgio/jncc2.html

[4] http://archive.ics.uci.edu/ml/

6 Tree Augmented Naive Credal Classifier

The Tree-Augmented Naive (TAN) structure has the characteristic that each feature has at least C as parent and at most one other parent constituted by another feature. By Tree Augmented Naive Credal Classifier (TANC), we mean a credal network over a TAN graph.

As described in Section 4, TANC performs pairwise comparison to detect credal-dominance; for every comparison between two classes, the minimization is performed over (a) all possible completions of the training data (because missing data of the training set are nonMAR) and (b) over the prior densities belonging to the EDM. The credal dominance condition can be rewritten as:

$$\min_{\mathbf{d}_{\mathcal{X}},\boldsymbol{\alpha}} (p(c'|\mathbf{y}) - p(c''|\mathbf{y})) > 0,$$

because the distributions $p \in \mathcal{K}$ are completely defined by $\mathbf{d}_{\mathcal{X}}$ and $\boldsymbol{\alpha}$.

We assume further that there is no missing values in the class and that the hyper-parameters $\boldsymbol{\alpha}_C$ are fixed (we may solve at each time a given extreme configuration of $\boldsymbol{\alpha}_C$). Hence, the credal dominance problem is equivalent to

$$\min_{\mathbf{d}_{\mathcal{X}},\boldsymbol{\alpha}} (p(\mathbf{y}|c')p(c') - p(\mathbf{y}|c'')p(c'')) > 0$$

because $p(\mathbf{y})$ is positive and so does not affect the sign of the formula. Then we can separately solve each optimization as follows:

$$p(c') \cdot \min_{\mathbf{d}_{\mathcal{X}},\boldsymbol{\alpha}\setminus\boldsymbol{\alpha}_C} p(\mathbf{y}|c') - p(c'') \cdot \max_{\mathbf{d}_{\mathcal{X}},\boldsymbol{\alpha}\setminus\boldsymbol{\alpha}_C} p(\mathbf{y}|c'') \quad (4)$$

because $p(\mathbf{y}|c')$ only depends on $\alpha_{c'}$ and on the data of instances with $C = c'$, while $p(\mathbf{y}|c'')$ depends on $\alpha_{c''}$ and counts from instances with $C = c''$ (data with $C = c'$ and $C = c''$ are obviously disjoint).

Because we take the Extreme IDM as model for the priors, $\boldsymbol{\alpha}$ only assumes extreme values. Hence, it is possible to tackle the problem by introducing s new instances to the training set that are completely missing. As this new *fake* instance of missing values has also missing classes, it could introduce a dependence between the minimization and the maximization of Equation (4). However, it is possible to solve the optimization for every possible completion of the missing data of the class in this additional instance (which are just two extremes). Thus we have

$$p(c') \cdot \min_{\mathbf{d}_{\mathcal{X}}} p(\mathbf{y}|c') - p(c'') \cdot \max_{\mathbf{d}_{\mathcal{X}}} p(\mathbf{y}|c''), \quad (5)$$

which is solved for every possible completion of the data (including the fake instance).

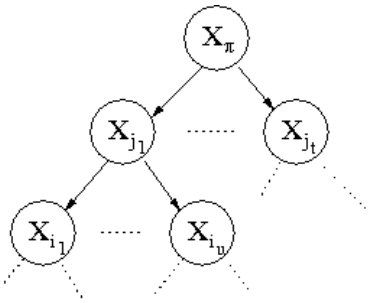

Figure 2: Part of the computation tree of the TANC algorithm.

The idea of the algorithm to evaluate Equation (5) is to combine the computations that are performed separately in the children of each variable and then to propagate the best possible solution to their sole parent. We ignore the arcs from C because we look for $\underline{p}(\mathbf{y}|c') = \min_{\mathbf{d}_\mathcal{X}} p(\mathbf{y}|c')$ and $\overline{p}(\mathbf{y}|c'') = \max_{\mathbf{d}_\mathcal{X}} p(\mathbf{y}|c'')$, that is, the actual root variable C is observed. The computation starts on the leaves and follows in a bottom-up idea. At each variable X_i, the goal is to obtain the joint probability $p(\mathbf{y}_{\mathbf{\Lambda_i}}|y_i, c)$ of its children conditional on y_i[5] (c equals c' or c'' depending whether it is the minimization or the maximization). This evaluation is done for all possible completions d_{X_i} and it is optimized over the completions of the children. The result is stored in a *cache* $\phi_i(d_{x_i})$. Figure 2 shows part of a network. At X_{j_1}, the joint probabilities $p(\mathbf{y}_{\mathbf{\Lambda_{i_k}}}|y_{i_k}, c)$ of every child $X_{i_k} \in \Lambda_{j_1}$ (for every possible completion of that sub-tree) are already computed. So, they are combined to obtain $p(y_{i_1}, \ldots, y_{i_u}|y_{j_1}, c)$, for every possible completion of X_{j_1}. These new probabilities $p(\mathbf{y}_{\mathbf{\Lambda_{j_1}}}|y_{j_1}, c)$ are then made available to the parent X_π, where the computations are analogous but using the information obtained from X_{j_1} and its siblings. The process goes through the tree structure until reaching the root.

Denote by $\mathbf{y}_{\sigma(\mathbf{i})}$ the assignment for all the variables in the sub-tree rooted at X_i, that is, $\mathbf{y}_{\sigma(\mathbf{i})} \in \Omega_{\mathbf{X}_{\sigma(\mathbf{i})}}$ is the queried assignment over $\mathbf{X}_{\sigma(\mathbf{i})} \subseteq \mathcal{X}$, the set of variables in the sub-tree rooted at X_i. Suppose that the root variables (if C is not considered) are $X_1, \ldots, X_r$. So,

$$\begin{aligned} p(\mathbf{y}|c') &= \prod_{j=1}^{r} p(\mathbf{y}_{\sigma(\mathbf{j})}|c') \\ &= \prod_{j=1}^{r} p(y_j|c') \cdot \prod_{X_i \in \Lambda_j} p(\mathbf{y}_{\sigma(\mathbf{i})}|y_j, c'), \end{aligned}$$

and, in general, $\min_{\mathbf{d}_\mathcal{X}} p(\mathbf{y}_{\sigma(\mathbf{j})}|\pi_j^y, c') =$

$$= \min_{\mathbf{d}_\mathcal{X}} \left(p(y_j|\pi_j^y, c') \cdot \prod_{X_i \in \Lambda_j} p(\mathbf{y}_{\sigma(\mathbf{i})}|y_j, c') \right),$$

[5] $y_i \in \Omega_{X_i}$ is used as the notation for the queried state of X_i.

where $\pi_j^y \in \Omega_{\Pi_j}$ is the assignment of Π_j that is being queried. (the maximization is analogous). Now, when you complete the variable X_j, the children Λ_j have separable computations. They are separable because the counts n that appear in the children of X_j are independent of each other as they concern disjoint subsets of variables (the structure is a tree, so $\mathbf{X}_{\sigma(\mathbf{i})} \cap \mathbf{X}_{\sigma(\mathbf{i}')} = \emptyset$ for $X_i, X_{i'} \in \Lambda_j$, with $i \neq i'$ and $X_j = \Pi_i = \Pi_{i'}$.). The only dependent value is n_{y_j}, as it appears in the denominators of distinct children of X_j. However, n_{y_j} is fixed as the problem is solved for every possible completion of X_j. Besides that, note that the terms $\boldsymbol{\alpha}$ are not present because we treat them using the *fake* missing instance. Hence, the overall computation can be decomposed as

$$= \min_{d_{X_j}} \left(p(y_j|\pi_j^y, c') \cdot \prod_{X_i \in \Lambda_j} \min_{d_{\mathbf{X}_{\sigma(\mathbf{i})}}} p(\mathbf{y}_{\sigma(\mathbf{i})}|y_j, c') \right).$$

To prove that this idea is correct, we rewrite it as a function of completions: $\forall d_{\mathbf{X}_{\sigma(\mathbf{j})}}$, we have

$$\phi_j(d_{\mathbf{X}_{\sigma(\mathbf{j})}}) = \prod_{X_i \in \Lambda_j} \min_{d_{\mathbf{X}_{\sigma(\mathbf{i})}}} \left(\frac{n_{y_i y_j}}{n_{y_j}} \phi_i(d_{\mathbf{X}_{\sigma(\mathbf{i})}}) \right), \quad (6)$$

where the product is assumed to be 1 when Λ_j is empty. The maximization version is analogous. We prove by induction on the tree the following property:

$$\phi_j(d_{\mathbf{X}_{\sigma(\mathbf{j})}}) = \begin{cases} 1, & \text{if } X_j \text{ is a leaf,} \\ \underline{p}(\mathbf{y}_{\sigma(\mathbf{j})} \setminus \{y_j\}|y_j, c'), & \text{otherwise.} \end{cases} \quad (7)$$

The base of induction holds by definition. Now assume that Equation (7) holds for every $X_i \in \Lambda_j$. By applying this hypothesis on Equation (6), we have

$$\phi_j(d_{\mathbf{X}_{\sigma(\mathbf{j})}}) = \prod_{X_i \in \Lambda_j} \min_{d_{\mathbf{X}_{\sigma(\mathbf{i})}}} \left(\frac{n_{y_i y_j}}{n_{y_j}} \underline{p}(\mathbf{y}_{\sigma(\mathbf{i})} \setminus \{y_i\}|y_i, c') \right), \quad (8)$$

where n_{y_j} is fixed and $n_{y_i y_j}$ depends on the completion d_{X_i}, which belongs to $d_{\mathbf{X}_{\sigma(\mathbf{i})}}$. Thus, it is possible to minimize the factor of each child separately and we obtain $\phi_j(d_{\mathbf{X}_{\sigma(\mathbf{j})}}) = \underline{p}(\mathbf{y}_{\sigma(\mathbf{j})} \setminus \{y_j\}|y_j, c')$.

The derivation so far requires exponential time over all missing values. Nevertheless, an important fact in Equation (6) is that $\phi_i(d_{\mathbf{X}_{\sigma(\mathbf{i})}}) = \phi_i(d_{X_i})$, for d_{X_i} compatible with $d_{\mathbf{X}_{\sigma(\mathbf{i})}}$, that is, it is enough to keep the best possible solution for every completion of a variable without having to record all the completions of its descendants. This is valid because $n_{y_i y_j}$ is known when the completion d_{X_i} is given, so completions of variables in $\mathbf{X}_{\sigma(\mathbf{i})} \setminus \{X_i\}$ are irrelevant for the minimization in Equation (6), and it is enough to have the best possible solution for each d_{X_i}. This leads us only to compute:

$$\forall_{d_{X_j}} \ \phi_j(d_{X_j}) = \prod_{X_i \in \Lambda_j} \min_{d_{X_i}} \left(\frac{n_{y_i y_j}}{n_{y_j}} \phi_i(d_{X_i}) \right), \quad (9)$$

and equivalently in the maximization case. Now the algorithm can be implemented in a bottom-up manner so as the ϕ's of children are available when a given variable is treated, which reduces the complexity of the method to be exponential in the number of missing values of only two variables (a variable and its parent) instead of all missing values.

The described formulation obtains $\overline{p}(\mathbf{y} \setminus \{y_i\}|c'')$ and $\underline{p}(\mathbf{y} \setminus \{y_i\}|c')$, for each root variable X_i, $i \leq r$. Those values still need to be multiplied by the corresponding $p(y_i|c)$ (using the proper c). We leave this last step intentionally apart to show how to deal with the forest of trees. The probability of the variables that have only C as parent are multiplied all together, just as if we had computed $\phi_C(\cdot)$ according to Equation (9):

$$\underline{p}(\mathbf{y}|c') = \phi_C(\cdot) = \prod_{X_i \in \Lambda_C} \min_{d_{X_i}} \left(\frac{n_{y_i c'}}{n_{c'}} \phi_i(d_{X_i}) \right), \quad (10)$$

and similarly for the maximization. In case $r = 1$ (single root), the outer product of Equation (10) disappears. This final step returns the desired values $\overline{p}(\mathbf{y}|c'')$ and $\underline{p}(\mathbf{y}|c')$, which are later multiplied by $p(c'')$ and $p(c')$, respectively, to evaluate Equation (5).

We point out that, if the data set is complete, the only missing data that must be processed by the algorithm are those introduced by the fake instance (for the treatment of the EDM). In such case, the complexity of the method is clearly linear in the input size, as there is a constant number of computations by variable (there are only two ways of completing the data by variable and the algorithm is locally exponential). In fact there are other ideas that might be employed to solve the problem of selecting the hyperparameters $\boldsymbol{\alpha}$ of the EDM, but we use the idea of *fake* instance because it fits straightforward into the framework of the proposed algorithm. In the presence of missing data, the idea spends exponential time in the number of missing data of two linked variables, which is already much better than an overall exponential but still slow for data sets with many missing values. Using dynamic programming, it might be possible to further reduce this complexity to exponential in the missing of a single variable.

7 Experiments on TAN

We have performed experiments on several data sets retrieved from the UCI repository. The data sets cover a wide spectrum in terms of number of instances (min: 101; max: 12960) and classes (min: 2; max: 11). On each data set, we have performed 10-folds cross-validation to the performance of the classifiers. Numerical features have been discretized using supervised discretization [4]; the features discretized into a single bin have been removed from the computation. TANC requires the features to be discrete; however, supervised discretization of the features is a good practice in general, as it has been shown to improve the accuracy of several classifiers [3].

The instances over which TANC return a single class are referred to as *determinately* classified, while those over which TANC returns more classes are referred to as *indeterminately* classified.

For some data sets, we report results before and after having performed feature selection. To perform feature selection, we have cross-checked the suggestions of two feature selectors implemented in WEKA [9]: correlation-based and wrapper. Both approaches are multivariate, i.e., they are designed to identify an optimal subset of feature, by also considering interaction between features.

7.1 TANC vs TAN

In this section, we compare TANC against TAN on *complete* data sets, i.e., with no missing data. On each cross-validation run, we first learn the structure of the graph using WEKA [9]; later, we run TAN and TANC, using the same network structure for both.

We adopt a set of indicators already known in literature [10] for comparing a credal and a Bayesian classifier; in particular:

- *determinacy (D%)*: the percentage of instances classified determinately by TANC;
- *TAN-D* and *TAN-I*: the accuracy of TAN on the instances which are classified determinately and indeterminately by TANC. As TANC is designed to separate hard-to-classify instances (that are prior dependent, and hence indeterminately classified) and easy-to-classify instances (those determinately classified), we shall observe TAN-D>TAN-I, because TAN-D is in fact the accuracy achieved both by TAN and TANC on the determinately classified instances. Actually, if TANC is determinate, TAN and TANC return the same classification (although the uniform prior adopted for TAN is not included in the credal set of the EDM, it empirically appears that if both the extreme priors of the EDM indicate the same class as the most probable one, TAN will lead to the same conclusion too).
- *set-accuracy (S-acc%)*: the ratio of the number indeterminate classifications which contain the actual class to the total number of indeterminate classifications;
- *indeterminate output size (ind. sz.)*: the average number of classes returned on the instances indeterminately classified.

Note that set-accuracy and indeterminate output size are meaningful only if the data set has more than two classes.

data set	#inst.	#cl.	D	TAN-D	TAN-I	S-acc	Ind. Sz.
zoo	101	7	77%	100%	84%	100%	5.9/7
iris	150	3	93%	97%	44%	92%	2.8/3
diabetes	767	2	98%	80%	9%	-	-
segment	810	7	17%	99%	93%	98%	4.1/7
vehicle	846	7	67%	82%	60%	88%	2.4/7
vowel	990	11	66%	98%	77%	99%	7.6/11
credit	1000	2	87%	78%	55%	-	-
splice	3190	3	90%	97%	79%	99%	2.1/3
kr-kp	3196	2	99%	92%	40%	-	-
waveform	5000	3	88%	85%	73%	100%	2.1/3
nursery	12960	5	90%	97%	79%	99%	3.5/5
average			**79%**	**91%**	**63%**	**97%**	**72%**

Table 1: The data sets (sorted according to the number of instances) and the indicators. The meaning of the header is as follows: #inst denotes the number of instances of the data set and #cl the number of classes; D is the determinacy of TANC; TAN-D and TAN-I the accuracy achieved by TAN on the instances classified determinately and indeterminately by TANC; S-acc is the set-accuracy of TANC while Ind. Sz. is the average number of instances returned by TANC on the instances indeterminately classified.

The results of Table 1 show that the determinacy of TANC is quite high: 79% on average, and often above 90%. In general, the determinacy of TANC increases with the number of instances in the data set (as large the data set as reduced the importance of the prior) and decreases, for similarly-sized data sets, with the number of classes. The major exception to this is segment (17 features, 7 classes, 810 instances); however, in this case feature selection can be helpful. It turns out that 10 out of the 17 features in segment are irrelevant; removing them from the data set and re-running the experiment increases the determinacy from 17% to 57%, with only a minor drop of accuracy on the instances determinately classified (TAN-D decreases from 99% on 17% of instances to 95% on 57% of instances).

Most importantly, TANC is quite effective in separating hard-to-classify from easy-to-classify instances. There is a sharp drop of accuracy of TAN when we move from determinate to indeterminate instances; on the average, the drop is about 28 percentage points. On data sets with two classes, the accuracy of TAN on the instances indeterminately classified is comparable to random guessing or even worse (diabetes: 9%; credit: 55%, kr-kp: 40%); however, as the number of classes increases, TAN performs better on the instances indeterminately classified (see for instance segment and zoo: TAN-I is 84% and 93% respectively). This might show that as the number of classes increases, TANC becomes indeterminate also on some instances that could be successfully classified. However, studies suggest that even this kind of problem can be significantly mitigated by feature selection [2]. As an example, let us consider the zoo data set, which has 15 features. By running feature selection, we find that there are 4 irrelevant features out of 15. Re-running the experiment on the pruned data, determinacy rises from 77% to 79%, TAN-D remains close to 100%, while TAN-I drops from 84% to 72%. Hence, in some particular data sets, feature selection can be helpful to improve the determinacy and/or the detection of hard-to-classify instances.

On the hard-to-classify instances, TANC preserves its reliability thanks to indeterminate classifications, providing set-accuracy close to 100%, while returning on the average about 70% of the total classes. All these findings are in good agreement with previous comparisons of Bayesian classifiers against their imprecise probability counterparts [2, 13].

7.2 TANC vs. TANC*

Two main differences exist between TANC and TANC* regarding the model of prior ignorance (TANC adopts the EDM, while TANC* adopts the local IDM) and the treatment of missing data in the training set (TANC* assumes MAR, while TANC does not). In this section, we focus on the impact of the models of prior ignorance on the two classifiers; in order to remove the effect of the treatment of missing data, we consider *complete* data sets. We did not implement TANC* in our code; rather, we have compared our results with those published in [13]. For this reason, the analysis tries to draw general conclusions rather than punctual ones. We consider here all the 6 complete data sets analyzed in [13].

On the basis of previous explanation, we can expect TANC to be more determinate that TANC*. However we have also to verify that it becomes determinate on instances that can be safely classified with a single class.

The comparison between the determinacy of TANC and TANC* is shown in the upper plot of Figure 3. On the

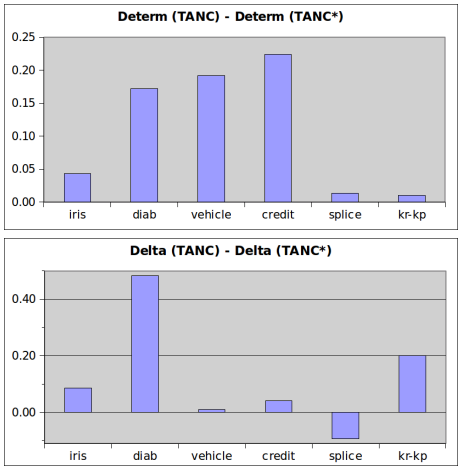

Figure 3: Comparison between TANC and TANC*.

average, TANC is 11 percentage points higher than that of TANC (89% vs. 78%). However, the determinacy of the two classifiers is almost equivalent on both splice and kr-kp; this might be due to the large size of the two data sets (around 3200 instances each), which reduces the role of the prior distributions.

In order to compare the ability of isolating hard-to-classify instances, we introduce the indicator Delta $=$ (TAN-D $-$ TAN-I), which evaluates the difference in accuracy achieved by TAN between the instances classified determinately and indeterminately by TANC [resp. TANC*]. The results are displayed in the lower plot of Figure 3; they suggests that the increased determinacy of TANC corresponds also to a better ability in isolating hard-to-classify instances, thus supporting the hypothesis that TANC is returning determinate answers on instances over which TANC* is unnecessarily indeterminate. However, these results should be taken with some cautiousness, as it has not been possible to actually run side-by-side the two classifiers.

7.3 Preliminary results with missing data

In this section we focus on comparing the determinacy of the classifier in the presence of missing data. The effect of the treatment of missing data is also important so as to verify the consequences of nonMAR in terms of accuracy, but a deeper analysis is left for future work. We note that the term nonMAR is employed to indicate the ignorance about the MP, that is, MAR is not assumed. In particular, we consider the crx data set, which has 16 features; the structure of the network has 14 links among features (besides those which connect the class to all the features). We consider the complete data set and then artificially generate 30 missing values, distributed among 6 different features. Even such a small quantity of missing data decreases the determinacy from 87% to 77%. On the very same data sets, we run the naive credal classifier 2 [2] which can be seen as NCC enabled for NonMAR treatment of missing data; the determinacy of NCC2 (assuming NonMAR) remains stable around 95% on both cases. Hence, it seems that the TAN structure can lead to much larger indeterminacy than the naive one, if MAR is not assumed. This result is somehow expected, as TAN introduces the possibility of having linked features with missing values, while a naive structure does not.

8 Conclusions

TANC is a new credal classifier based on a Tree-Augmented Naive structure; it treats missing data conservatively by considering all possible completions of the training set, but avoiding an exponential increase of the computational time. TANC adopt the EDM as a model of prior ignorance; we have shown that EDM is a reliable and computationally affordable model of prior near-ignorance for credal classifiers. We have shown that TANC is more reliable than precise TAN (learned with uniform prior) and that it obtains better performance compared to a previous TAN model based on imprecise probabilities, but learned with a local IDM approach; the adoption of EDM overcomes the problem of the unnecessary imprecision induced by the local IDM, while keeping the computation affordable.

The TANC classifier has room for many improvements. The treatment of MAR and nonMAR missing data all together, appearing both in the training and the testing set are the main topics for future work. In order to make TANC less indeterminate on incomplete data sets, a solution could be to allow for mixed configurations, in which some features are treated as MAR and some others are not. This would allow both for a decrease of indeterminacy and for a finer-grained tuning of the way that missing data are dealt with. Besides that, the computational performance of TANC can also be further improved, for example, with the use of dynamic programming. Extensions beyond trees are also of interest, but they fall into the need of fast and accurate inference methods for general credal networks.

Acknowledgments

Work partially supported by the Swiss NSF grant n. 200021-118071/1 and 200020-116674/1 and from the project 'Ticino in rete'.

References

[1] A. Cano, M. Gómez-Olmedo, and S. Moral. Credal nets with probabilities estimated with an extreme imprecise Dirichlet model. In *Proceedings of the Fifth International Symposium on Imprecise Probability:*

Theories and Applications (ISIPTA'07), Action M Agency, Prague, pages 57–66, 2007.

[2] G. Corani and M. Zaffalon. Learning Reliable Classifiers from Small or Incomplete Data Sets: the Naive Credal Classifier 2. *Journal of Machine Learning Research*, 9:581–621, 2008.

[3] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In A. Prieditis and S. Russell, editors, *Proceedings of the 12th conference on machine learning*, pages 194–202, San Francisco, CA, 1995. Morgan Kaufmann.

[4] U. M. Fayyad and K. B. Irani. Multi-interval Discretization of Continuous-valued Attributes for Classification Learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 1022–1027, San Francisco, CA, 1993. Morgan Kaufmann.

[5] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29(2):131–163, 1997.

[6] R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. Wiley, New York, 1987.

[7] P. Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, New York, 1991.

[8] P. Walley. Inferences from multinomial data: learning about a bag of marbles. *J. R. Statist. Soc. B*, 58(1):3–57, 1996.

[9] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques (Second Edition)*. Morgan Kaufmann, 2005.

[10] M. Zaffalon. Statistical inference of the naive credal classifier. In G. de Cooman, T. L. Fine, and T. Seidenfeld, editors, *ISIPTA '01: Proceedings of the Second International Symposium on Imprecise Probabilities and Their Applications*, pages 384–393, The Netherlands, 2001. Shaker.

[11] M. Zaffalon. Exact credal treatment of missing data. *Journal of Statistical Planning and Inference*, 105(1):105–122, 2002.

[12] M. Zaffalon. Conservative rules for predictive inference with incomplete data. In F. G. Cozman, R. Nau, and T. Seidenfeld, editors, *ISIPTA '05: Proceedings of the Fourth International Symposium on Imprecise Probabilities and Their Applications*, pages 406–415, Manno, Switzerland, 2005. SIPTA.

[13] M. Zaffalon and E. Fagiuoli. Tree-Based Credal Networks for Classification. *Reliable Computing*, 9(6):487–509, 2003.