Online active learning of decision trees with evidential data

Liyao Ma, Sébastien Destercke, Yong Wang

Online active learning of decision trees with evidential data

Liyao Ma[a,*], Sébastien Destercke[b], Yong Wang[a]

[a] *Department of Automation, University of Science and Technology of China, Hefei, China*
[b] *UMR7253 Heudiasyc, Centre de Recherches de Royallieu, Compiègne, France*

Abstract

Learning from uncertain data has been drawing increasing attention in recent years. In this paper, we propose a tree induction approach which can not only handle uncertain data, but also furthermore reduce epistemic uncertainty by querying the most valuable uncertain instances within the learning procedure. We extend classical decision trees to the framework of belief functions to deal with a variety of uncertainties in the data. In particular, we use entropy intervals extracted from the evidential likelihood to query selected uncertain querying training instances when needed, in order to improve the selection of the splitting attribute. Our experiments show the good performances of proposed active belief decision trees under different conditions.

Keywords: decision tree, active learning, evidential likelihood, uncertain data, belief functions

1. Introduction

Decision trees, as one of the best-known approaches for classification, are widely used due to their good learning capabilities and simplicity to understand. However, classical decision trees can only handle certain data whose values are precisely known. Those uncertain instances, despite the fact that they may

*Corresponding author
Email addresses: liyma@mail.ustc.edu.cn (Liyao Ma), sebastien.destercke@hds.utc.fr (Sébastien Destercke), yongwang@ustc.edu.cn (Yong Wang)

contain useful information, are usually ignored or removed by replacing them with precise instances when building decision trees [1], potentially leading to a loss of accuracy. Different approaches have been proposed to overcome this drawback, such as probabilistic decision trees developed by Quinlan [2] and Smith [3], fuzzy decision trees proposed by Yuan [4] and Wang [5] or uncertain decision trees proposed by Biao *et al.* [6] and Liang *et al.* [7].

The interest for integrating uncertain data in learning methods has been growing in the recent years [8, 9, 10, 11]. While probability theory is the most commonly used tool to model this uncertainty, various authors (see the special issue [12] and papers within it) have argued that probability cannot always adequately represent data uncertainty (often termed epistemic uncertainty). For instance, probabilistic modelling is unable to model faithfully set-valued observations. In this paper, we will work with a more general theory, the theory of belief functions (also called Dempster-Shafer theory or evidence theory) [13, 14], which has the advantage to include both sets and probabilities as specific cases.

Embedding belief function within the learning of decision trees has already been investigated in the past. Elouedi et al. [17, 18] discussed belief decision tree construction under TBM model. Vannoorenberghe [19, 20] concentrated on the aggregation of belief decision trees. Sutton-Charani et al. [21, 22] proposed to estimate tree parameters by maximizing evidential likelihood function using the E^2M algorithm [23].

However, although those proposals deal with uncertain data modelled by belief functions, none of them have looked at the issue of reducing data uncertainty through information querying. This is what we propose in this paper: to query uncertain data during the tree learning procedure, in order to improve its performances. In some sense, this idea is very close to the one of active learning [24], where the learning algorithm can achieve higher accuracies by actively selecting the most valuable unlabelled instances and querying their true labels. There are however two significant differences between our proposal and the usual active learning: we consider generic uncertain data (modelled by belief functions) and we query while learning the model (a process close to online

active learning [25]), rather than "learning then querying". To our knowledge, this paper is the first to propose an evidential online active learning method. We do this by relying on the notion of evidential likelihood [26] to select the items to query in order to improve the split selection.

Apart from the query method mentioned above, our proposal also allows us to derive an alternative approach to learning decision trees from uncertain data: considering not only the maximal likelihood value, we extract entropy intervals from the evidential likelihood and use these intervals to choose the best split at each decision node. The proposed approach includes more information about the likelihood, and extends both classical C4.5 decision trees and the E^2M decision trees [21] applied to uncertain outputs (but certain inputs).

Section 2 recalls the necessary material on belief functions, classical decision trees and evidential likelihood. In Section 3, we discuss in detail the tree induction procedure, including entropy interval generation, attribute selection, query strategy and the overall algorithm description. Section 4 details some experiments on classical UCI data sets, and compare the results of the proposed approach with decision trees without querying. Finally, conclusions are given in Section 5.

2. Settings and basic definitions

The purpose of a classification approach is to build a model $\mathscr{M}$ that maps a feature vector $\boldsymbol{x} = (x^1, \ldots, x^k) \in A^1 \times A^2 \times \ldots \times A^k$ taking its values on k attributes, to an output class $y \in \mathscr{C} = \{C_1, \ldots, C_\ell\}$ taking its value among ℓ classes. Each attribute $A^i = \{a^i_1, \ldots, a^i_{r_i}\}$ has r_i possible values. This model is then used to make predictions on new instances $\boldsymbol{x}$ whose classes are unknown. Typically this model $\mathscr{M}$ (a decision tree, a Bayes network, a logistic regression, ...) is learned from a training set of precise data, denoted as

$$T = \begin{pmatrix} \boldsymbol{x_1}, y_1 \\ \vdots \\ \boldsymbol{x_n}, y_n \end{pmatrix} = \begin{pmatrix} x^1_1, \cdots, x^k_1, y_1 \\ \vdots \\ x^1_n, \cdots, x^k_n, y_n \end{pmatrix}.$$

However, in practical applications, it is possible to have uncertainty in both inputs (feature vectors) and outputs (classification labels). This uncertainty is epistemic[1], in the sense that a given x_j^i or y_i has a unique true value that may be ill-known. As recalled in the introduction, the adequacy of probability theory to model such uncertainty is questionable, hence in this paper we will model uncertainty by belief functions. Also, we will consider the case where the input is certain and only the output is uncertain (a classical assumption in active learning).

2.1. Belief functions

Let a finite space $\mathscr{C}$ be the frame of discernment containing all the possible exclusive values that a variable (here, the output class y) can take. When the true value of y is ill-known, our uncertainty about it can be modelled by a mass function $m_y : 2^{\mathscr{C}} \rightarrow [0, 1]$, such that $m_y(\emptyset) = 0$ and

$$\sum_{E \subseteq \mathscr{C}} m_y(E) = 1. \tag{1}$$

A subset E of $\mathscr{C}$ is called a *focal set* of m_y if $m_y(E) > 0$. $m_y(E)$ can then be interpreted as the amount of evidence indicating that the true value is in E. The following typical mass functions show that this model extends both set-valued and probabilistic uncertainty models:

- a *vacuous* mass is such that $m_y(\mathscr{C}) = 1$. It represents total ignorance;
- a *Bayesian* mass is such that $m_y(E) > 0$ iff $|E| = 1$. It is equivalent to a probability distribution;
- a *logical (categorical)* mass is such that $m_y(E) = 1$ for some E. It is equivalent to the set E.

[1] by opposition of so-called aleatory uncertainty, which concerns a stochastic behaviour.

The associated belief and plausibility functions, which are in one-to-one relations with the mass function m_y, are defined as:

$$Bel_y(B) = \sum_{E \subseteq B} m_y(E), \tag{2}$$

$$Pl_y(B) = \sum_{E \cap B \neq \emptyset} m_y(E), \tag{3}$$

for all $B \subseteq \mathscr{C}$. The belief function measures how much event B is certain (it sums masses implying B), while the plausibility measures how much event B is consistent with available evidence. The function $pl_y : \mathscr{C} \to [0, 1]$ such that $pl_y(w) = Pl_y(\{w\})$ is called the contour function associated to m_y.

When modelling uncertain outputs by mass functions, the training set becomes

$$T = \begin{pmatrix} \boldsymbol{x_1}, m_{y_1} \\ \vdots \\ \boldsymbol{x_n}, m_{y_n} \end{pmatrix} = \begin{pmatrix} x_1^1, \cdots, x_1^k, m_{y_1} \\ \vdots \\ x_n^1, \cdots, x_n^k, m_{y_n} \end{pmatrix}.$$

2.2. Decision trees

Decision trees [27] are commonly used classifiers that induce a rooted tree structure, in which leaves (the terminal nodes) represent class labels and branches represent features with associated values leading to the nodes.

To be able to predict to which of ℓ classes belong an instance with k attributes, decision trees are induced top-down from a training set T. Every decision node (non-terminal node) is associated with a splitting attribute, which is selected by an attribute selection strategy, that can be based on different algorithms and purity measures [28]. The splitting process is then repeated recursively until a stopping criterion is met. The achieved decision tree then determines a partition of the instance space, and associates a class to each element of this partition. This means that each terminal node (or leaf) of a decision tree can be associated to an element of the partition. Figure 1 shows a classical decision tree and the associated partition when $k = 2$ and $\mathcal{C} = \{a, b, c\}$.

Several algorithms have been proposed for decision tree learning, among which ID3 [27], C4.5 [29] and CART [30] are the most commonly used. In this

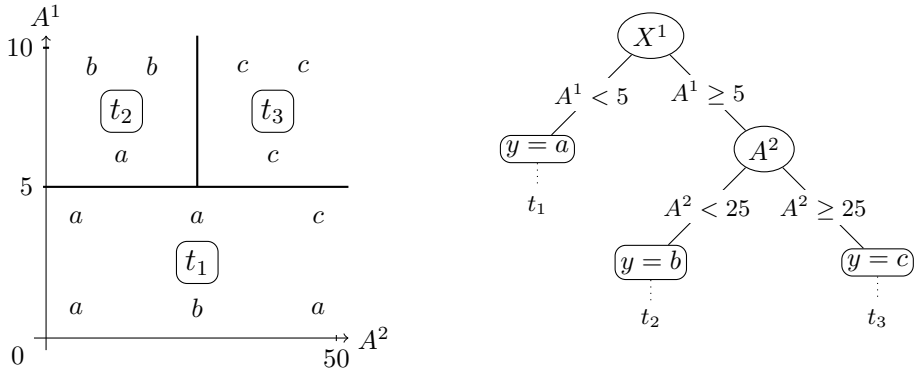


Figure 1: Example of decision tree and associated partition

paper, we take the basic C4.5 algorithm and entropy as an information measure to compute and evaluate the quality of a node split by a given attribute. Our querying strategy can easily be extended to other algorithms or information measures, yet as our main goal is to check that our approach improves upon the ones not considering and not querying uncertain data, we prefer to consider a well-known and simple induction method.

For the set of possible labels $\mathscr{C} = \{C_1, \ldots, C_\ell\}$, we associate to the ℓ classes fractions θ_i with $\sum_{i=1}^{\ell} \theta_i = 1$, where θ_i is the estimated probability of class C_i. Possible parameters $\boldsymbol{\theta}$ form a parameter space $\Theta = \{(\theta_1, \ldots, \theta_\ell) \mid \sum_{i=1}^{\ell} \theta_i = 1, 0 \leq \theta_i \leq 1, i = 1, \ldots, l\}$. Given a node and the set T of instances belonging to the partition element associated to this node (in the rest of the paper, we will simply say "belonging to this node"), θ_i in classical decision trees is the proportion of instances in T that are of class C_i. The formula of entropy is then

$$Info(T) = -\sum_{i=1}^{\ell} \theta_i \log_2(\theta_i). \tag{4}$$

Given an attribute A^k having r_k modalities, its gain ratio is defined as

$$Gain\ ratio(T, A^k) = \frac{Gain(T, A^k)}{Split\ Info(T, A^k)}, \tag{5}$$

where

$$Gain(T, A^k) = Info(T) - Info_{A^k}(T) \tag{6}$$

$$Info_{A^k}(T) = \sum_{i=1}^{r_k} \frac{\mid T_i \mid}{\mid T \mid} Info(T_i) \tag{7}$$

$$Split\ Info(T, A^k) = -\sum_{i=1}^{r_k} \frac{\mid T_i \mid}{\mid T \mid} \log_2 \frac{\mid T_i \mid}{\mid T \mid} \tag{8}$$

with $\mid T \mid$ and $\mid T_i \mid$ the cardinalities of the instance sets belonging to a parent node and to the child node i corresponding to $A^k = a_i^k$ (i.e. instances in the parent node such that $x^k = a_i^k$), respectively. In C4.5, $Split\ Info$ ensures that attributes having a higher number of modalities will not be favoured unduly during the splitting selections.

The attribute with the largest gain ratio is then selected for splitting, and a new child node is generated for each of its value, dividing the instance space into several subspaces. Once the tree is induced, a new instance can be classified by starting at the root node and moving down tree branches according to its feature values until a leaf node is reached. The predicted label is the class that is in majority among instances in the leaf. The accuracy of a decision tree can then be evaluated on a test set by comparing the predicted class labels with the true observed labels.

2.3. Evidential likelihood

When the observations are uncertain and modelled by m_y, it is no longer possible to evaluate proportions θ_i through empirical frequencies. One alternative [26] is to use the statistical tool of evidential likelihood to perform parameter estimation. Let Y be a discrete random variable taking value on $\mathscr{C}$ and y be a realization of Y. Given a parametric model $p_Y(\cdot; \boldsymbol{\theta})$ with parameter vector $\boldsymbol{\theta} \in \Theta$, the likelihood of $\boldsymbol{\theta}$ for a perfect observation y is defined as $L(\boldsymbol{\theta}; y) = p_Y(y; \boldsymbol{\theta}),\ \forall \boldsymbol{\theta} \in \Theta$. If y is imprecisely observed (we only know $y \in B$), the imprecise likelihood of $\boldsymbol{\theta}$ is described as

$$L(\boldsymbol{\theta}; B) = p_Y(B; \boldsymbol{\theta}) = \sum_{y \in B} p_Y(y; \boldsymbol{\theta}) \quad \forall \boldsymbol{\theta} \in \Theta \tag{9}$$

Furthermore, when the observation m_y is both imprecise and uncertain, the evidential likelihood can be defined as [31]

$$\begin{aligned} L(\boldsymbol{\theta}; m_y) &= \sum_{B \subseteq \mathscr{C}} L(\boldsymbol{\theta}; B) m_y(B) \\ &= \sum_{\omega \in \mathscr{C}} p_Y(\omega; \boldsymbol{\theta}) \sum_{B \ni \omega} m_y(B) \\ &= \sum_{\omega \in \mathscr{C}} p_Y(\omega; \boldsymbol{\theta}) pl_y(\omega) \quad \forall \boldsymbol{\theta} \in \Theta \end{aligned} \tag{10}$$

As $L(\boldsymbol{\theta}; m_y)$ only depends on the contour function pl_y induced by m_y, it can be written as $L(\boldsymbol{\theta}; pl_y)$ instead. From (10), we have $L(\boldsymbol{\theta}; pl_y) = \mathbb{E}_{\boldsymbol{\theta}}[pl_y(Y)]$.

If we observe a set of cognitively independent (see Denoeux [31] for a definition of cognitively independent) and i.i.d. uncertain observations $\boldsymbol{y} = (y_1, \ldots, y_n)$, where every observation y_i is modelled by m_{y_i}, the evidential likelihood of the corresponding multinomial distribution (having θ_j as parameters) becomes

$$L(\boldsymbol{\theta}; m_{\boldsymbol{y}}) = \mathbb{E}_{\boldsymbol{\theta}}[pl_{\boldsymbol{y}}(Y)] = \prod_{i=1}^{n} \mathbb{E}_{\boldsymbol{\theta}}[pl_i(Y)] = \prod_{i=1}^{n} \sum_{j=1}^{\ell} \theta_j pl_i(j), \tag{11}$$

where we denote $pl_i(j) := pl_{y_i}(C_j)$ for brevity's sake. The corresponding contour function (an extension of the so-called relative likelihood [32]) is given by

$$pl_{\Theta}(\boldsymbol{\theta}; m_{\boldsymbol{y}}) = \frac{L(\boldsymbol{\theta}; m_{\boldsymbol{y}})}{\sup_{\theta \in \Theta} L(\boldsymbol{\theta}; m_{\boldsymbol{y}})} \tag{12}$$

3. Induction of active belief decision trees

Up to now, there have been various proposals to deal with evidential data and decision tree induction (see Introduction), yet all of them take data uncertainty as a given and do not try to reduce it. Yet, attempting to reduce data uncertainty (especially as it is epistemic) can improve the model accuracy, as witnessed by some recent works focusing on fuzzy data [10] (which can be seen as a special case of belief functions). Our goal here is to propose a method to perform data querying (asking the true labels of uncertain data) while learning, using at our advantage the properties of the evidential likelihood induced by the contour function. We also do so only when we need, trying to select those

data that will be the most helpful to select a good split. To our knowledge, this is the first proposal within evidence theory that goes in this sense. A closely related recent proposal is the one of Reineking and Schill [33], yet it uses the notion of Pignistic transform and is not applied to trees.

This section explains the proposed method, which is based on calculating entropy intervals from the evidential likelihood, and on querying data when these intervals are insufficient to chose the best attribute to split on.

3.1. Generation of entropy intervals

To generate the entropy intervals from data m_y that are within a node, we will estimate $\boldsymbol{\theta}$ by using the evidential likelihood induced from the multinomial distribution (Equation (12)). These estimations will then be mapped to entropy intervals.

Definition 1 (α-cut of Θ)**.** Let $\Theta = \{(\theta_1, \ldots, \theta_\ell) \mid \sum_{i=1}^{\ell} \theta_i = 1, 0 \leq \theta_i \leq 1, i = 1, \ldots, l\}$ be the parameter set with a contour function $pl : \Theta \to [0, 1]$. Given a real number $\alpha \in [0, 1]$, the α-cut of Θ is denoted as

$$L_\alpha := \{\boldsymbol{\theta} \in \Theta \mid pl_\Theta(\boldsymbol{\theta}; m_y) \geq \alpha\} \tag{13}$$

The contour function pl_Θ shown in formula (12) is a concave function (this is proved in Appendix A). Therefore the α-cut L_α is always a convex set of Θ, which means that all possible entropies within L_α form a closed interval over the real numbers $\mathbb{R}$, as seen in Definition 2.

Definition 2 (entropy interval)**.** For an α-cut of Θ, the entropy interval which contains all the possibly-taken entropy values is denoted by

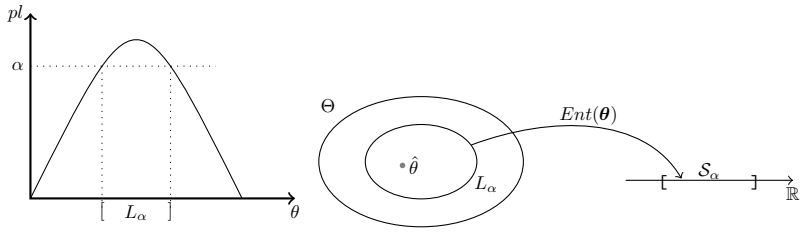
$$\mathcal{S}_\alpha := \{Ent(\boldsymbol{\theta}) \mid \boldsymbol{\theta} \in L_\alpha\} = [\underline{Ent_\alpha}, \overline{Ent_\alpha}], \tag{14}$$

with respectively

$$\underline{Ent_\alpha} = \inf_{\boldsymbol{\theta} \in L_\alpha} Ent(\boldsymbol{\theta}), \quad \overline{Ent_\alpha} = \sup_{\boldsymbol{\theta} \in L_\alpha} Ent(\boldsymbol{\theta})$$

the lower and upper bounds of entropy $Ent(\boldsymbol{\theta}) = -\sum_{i=1}^{\ell} \theta_i \log_2(\theta_i)$.

The passage from the evidential likelihood to entropy intervals is illustrated in Figure 2.



(a) illustration of α-cut of Θ (b) mapping from space Θ to entropy interval $\mathcal{S}_\alpha$ in $\mathbb{R}$

Figure 2: Mapping relation of contour function pl_Θ, α-cut L_α and entropy interval $\mathcal{S}_\alpha$

Definition 3. The width of an entropy interval $\mathcal{S}_\alpha$ is given by

$$w(\mathcal{S}_\alpha) := \overline{Ent_\alpha} - \underline{Ent_\alpha} \tag{15}$$

For a parameter set $\Theta = \{(\theta_1, \ldots, \theta_\ell) \mid \sum_{i=1}^{\ell} \theta_i = 1, 0 \leq \theta_i \leq 1, i = 1, \ldots, \ell\}$, we have $w(\mathcal{S}_\alpha) \in [0, max(Ent)]$, where $max(Ent) = -log_2(\frac{1}{\ell}) = log_2(\ell)$. Note that $w(\mathcal{S}_\alpha)$ is a decreasing function of α, i.e., it is minimal for $\alpha = 1$ and maximal for $\alpha = 0$. In particular, if $\alpha = 1$, the interval $\mathcal{S}_1(T)$ will be a unique value, unless all instances in T are missing or imprecise.

Thanks to these tools, we are now able to calculate entropy values for every decision node of the tree, even when having uncertain observations of the classes. The estimation is now an interval rather than a crisp value, the width $w(\mathcal{S}_\alpha)$ of this interval measuring our uncertainty about the information measure of the node.

3.2. Attribute selection and splitting strategy

As in C4.5 decision trees, we propose to use information gain ratio to select the best split. However, this value now becomes interval-valued, as each node is associated with an entropy interval $\mathcal{S}_\alpha$. Given an attribute A^k with r_k

modalities, the information gain ratio (7) becomes

$$
\begin{aligned}
IG_\alpha(A^k) &= \frac{Gain(T, A^k)}{Split\ Info(T, A^k)} \\
&= [\frac{\underline{Ent_\alpha}(T) - \sum_{i=1}^{r_k} \frac{|T_i|}{|T|} \overline{Ent_\alpha}(T_i)}{Split\ Info(T, A^k)},\ \frac{\overline{Ent_\alpha}(T) - \sum_{i=1}^{r_k} \frac{|T_i|}{|T|} \underline{Ent_\alpha}(T_i)}{Split\ Info(T, A^k)}] \\
&:= [\underline{IG_\alpha}(A^k),\ \overline{IG_\alpha}(A^k)] \qquad (16)
\end{aligned}
$$

with

$$Gain(T, A^k) = \mathcal{S}_\alpha(T) - \mathcal{S}_\alpha^{A^k}(T) \qquad (17)$$

$$\mathcal{S}_\alpha^{A^k}(T) = \sum_{i=1}^{r_k} \frac{\mid T_i \mid}{\mid T \mid} \mathcal{S}_\alpha(T_i) \qquad (18)$$

$$Split\ Info(T, A^k) = -\sum_{i=1}^{r_k} \frac{\mid T_i \mid}{\mid T \mid} \log_2 \frac{\mid T_i \mid}{\mid T \mid} \qquad (19)$$

where T and T_i remain the instance set within a parent node and child node corresponding to $A^k = a_i^k$. Note that their cardinalities are precise (as is $Split\ Info(T, A^k)$), because input values are assumed to be certain. $\mathcal{S}_\alpha(T)$ is the entropy interval computed from the instance set T.

Normally, we would select the attribute with largest information gain ratio, yet since they are now intervals, such an attribute is not necessarily uniquely defined. To compare the gain ratio intervals of candidate attributes, we give the definition of attribute dominance.

Definition 4 (attribute dominance)**.** Given two attributes A^i, A^j and their associated gain ratio intervals $IG_\alpha(A^i)$, $IG_\alpha(A^j)$, we say that attribute A^j dominates A^i (which we denote by $A^j \succ_I A^i$) if $\underline{IG_\alpha}(A^j) > \overline{IG_\alpha}(A^i)$.

It is obvious that the dominance relation is transitive, i.e., if $A^j \succ_I A^i$ and $A^i \succ_I A^h$, then $A^j \succ_I A^h$. For a set of attributes $A^1, \ldots, A^k$, when A^j dominates all the other attributes, i.e., $\exists A^j\ s.t.\ A^j \succ_I A^i, \forall A^i \neq A^j, i, j = 1, \ldots, k$, it will have the largest gain ratio undoubtedly, and we will call it the *dominant* attribute.

Given a decision node with an attribute set of k candidates $\mathscr{A} = \{A^1, ..., A^k\}$, if there is no dominant attribute, we have two options: trying to reduce the in-

tervals width of non-dominated attributes, to be able to compare them according to Definition 4, or try to propose some attribute selection relying on the entropy intervals. We therefore propose the following **attribute selection strategy**:

- If a dominant attribute A^j exists, it is selected for splitting;
- if

$$\mathscr{A}^q = \{A^i \in \mathscr{A} : \nexists A^j \ s.t. \ A^j \succ_I A^i\},$$

the set of non-dominated attribute according to our knowledge, has more than one element ($|\mathscr{A}^q| > 1$), then

  - if we can query some uncertain elements, then query (see Section 3.3) to refine $\mathscr{A}^q$, until $|\mathscr{A}^q| = 1$ (there is a dominant attribute) or we can no longer query (no uncertain instances are available, query request refused manually, maximal number of queries reached, etc.).
  - if we cannot query but still have multiple non-dominated elements ($|\mathscr{A}^q| > 1$), select the attribute with maximum mid-value of interval:

$$\arg\max_{A^j \in \mathscr{A}^q} \frac{\underline{IG_\alpha}(A^j) + \overline{IG_\alpha}(A^j)}{2}. \tag{20}$$

Example 1. For instance, if we have three attributes A^1, A^2, A^3 on which we can split, and given the chosen α, we have

$$IG_\alpha(A^1) = [0.7, 1.2], \quad IG_\alpha(A^2) = [0.5, 1.3], \quad IG_\alpha(A^3) = [0.3, 4]$$

then we can certainly eliminate A^3 from the possible splits ($\overline{IG_\alpha}(A^3) < \underline{IG_\alpha}(A^2)$), but not A^1 or A^2, since neither dominates the other. We therefore have $\mathscr{A}^q = \{A^1, A^2\}$, and either we can query to reduce the intervals $IG_\alpha(A^1), IG_\alpha(A^2)$, or if we cannot query we pick A^1 (since its middle point, 0.95, is higher than the one for A^2).

Having selected the best attribute A^j with r_j possible values, the decision node is split into r_j child nodes, while the instance set T is partitioned into r_j mutually exclusive subsets $T_1, ..., T_{r_j}$ according to values that attribute A^j takes.

Algorithm 1 summarizes the process of attribute selection. This algorithm is used iteratively to split instance space $A^1 \times \ldots \times A^k$ into more informative subspaces until a stopping criterion (minimal number of instances in a node reached, maximal tree depth reached, ...) is met.

Algorithm 1: Algorithm of attribute selection and splitting

Input: set of possible splitting attributes $\mathscr{A} = \{A^1, ..., A^k\}$, instance set T, maximal number of queries N_q, number of queries num_q

Output: selected attribute A^j, instance sets for all child nodes T_i, $i = 1, \ldots, r_j$, number of queries num_q

1 compute gain ratio interval $[\underline{IG_\alpha}(A^i), \overline{IG_\alpha}(A^i)]$ for each A^i;
2 update $\mathscr{A}$ into $\mathscr{A}^q = \{A^i \in \mathscr{A} : \nexists A^j \ s.t. \ A^j \succ_I A^i\}$;
3 **if** $|\mathscr{A}^q| > 1$ **then**
4 **if** $num_q < N_q$ **then**
5 query with Algorithm 2 and go to step 1
6 **else**
7 split on attribute $\arg\max_{A^j \in \mathscr{A}^q} \frac{\underline{IG_\alpha}(A^j) + \overline{IG_\alpha}(A^j)}{2}$.
8 **else**
9 split on $\mathscr{A}^q$

Example 2. Consider a training set with 21 training instances. Every instance has two attributes A^X (taking value in $\{a, b, c\}$) and A^Y (taking value in $\{d, e\}$) and is classified as one of three classes $\mathscr{C} = \{\bigcirc, \square, \times\}$. Representing the uncertain labels under belief function framework, the features, plausibility functions and true label of every instance are listed in Table 1.

There are two candidate attributes A^X and A^Y initially. When α=1, we get the precise gain ratios $IG_1(A^X) = 0.7631$ and $IG_1(A^Y) = 0.6933$, therefore attribute A^X would be selected as the best choice. If we now set $\alpha = 0.9$, the

Table 1: Uncertain instances modelled by belief functions

Number	Attributes		Plausibility functions			True
	X	Y	pl(○)	pl(□)	pl(×)	label
1	a	e	0.8	0.3	0.3	×
2	a	e	0.2	1	0.7	□
3	a	e	0.1	0.1	1	×
4	a	e	0.3	0	1	×
5	b	e	0.7	1	0.7	□
6	b	e	0.5	1	0	□
7	b	e	0.7	1	0.3	□
8	b	e	1	1	1	□
9	c	e	0	1	0	□
10	c	e	0	1	0	□
11	c	e	0	1	0	□
12	c	e	0.4	1	0	□
13	a	d	0.2	0.2	1	×
14	a	d	0	0	1	×
15	a	d	0.3	0	1	×
16	a	d	0	0	1	×
17	b	d	0.6	0.6	1	×
18	c	d	1	0	0	○
19	c	d	1	0.2	0	○
20	c	d	0.5	0.8	0.4	□
21	c	d	0.7	0.3	0	○

split on attribute A^X give the values

$$\begin{aligned}
\mathcal{S}_{0.9}(T) &= [1.4522, 1.5751] \\
\mathcal{S}_{0.9}^{A^X}(T) &= 8/21[0.0000, 0.1614] + 5/21[0.0000, 0.4822] + 8/21[0.7950, 1.0319] \\
&= [0.3029, 0.5694] \\
IG_{0.9}(A^X) &= [0.8828, 1.2722]/1.5538 = [0.5682, 0.8188].
\end{aligned}$$

The split on attribute A^Y give the values

$$\mathcal{S}_{0.9}(T) = [1.4522, 1.5751]$$
$$\mathcal{S}_{0.9}^{A^Y}(T) = 12/21[0.6098, 0.9285] + 9/21[0.9580, 1.1211] = [0.7591, 1.0110]$$
$$IG_{0.9}(A^Y) = [0.4412, 0.8161]/0.9852 = [0.4478, 0.8282].$$

Table 2 lists more results about the entropy and gain ratio intervals calculated with different α. As the intervals show, in all the three cases, no matter what value α takes, no dominant attribute exists. So we query at most 5 instances (according to process detailed in Section 3.3) to refine the results. The information gain ratios calculated after querying are denoted as $IG_\alpha^q(T, A^i)$. After querying, it can be seen that A^X dominates A^Y when α=0.9, therefore A^X can safely be selected for splitting. In the other two cases, we do not have $A^X \succ_I A^Y$, yet according to Equation (20), A^X is also selected.

The intervals typically become narrower after querying. Yet for lower values of α (such as $\alpha = 0.5$ in Table 2), querying has less effect on the narrowing of information gain intervals, which suggests that α value should not be too low during the learning process.

It can also happen that no dominant attribute is found even when all uncertain instances have been queried. For example, the results achieved by querying all uncertain instances are $IG_{0.8}(T, A^X) = [0.2540, 0.6044]$, $IG_{0.8}(T, A^Y) = [0.1715, 0.6226]$. In this case, we can use Equation (20) to select the best split. Note also that if intervals overlaps even when all instances have been queried, this gives us a natural stopping criterion (however, to use it, α should remain high).

3.3. Query strategy

As mentioned in the previous section, query is requested only when no dominant attribute exists. Each time a query is done, the selected uncertain instances are presented to an oracle (e.g., a human expert annotator, a reliable sensor) that provides the true label of the instance without uncertainty. This will tend to narrow entropy intervals and identify a dominant attribute. By querying, we

Table 2: Intervals achieved under different values of α

	α=0.5	α=0.8	α=0.9
$\mathcal{S}_\alpha(T)$	[1.2765, 1.5848]	[1.4033 , 1.5831]	[1.4522 , 1.5751]
$\mathcal{S}_\alpha(T(A^X = a))$	[0.0000 , 0.7051]	[0.0000 , 0.2864]	[0.0000 , 0.1614]
$\mathcal{S}_\alpha(T(A^X = b))$	[0.0000 , 1.4637]	[0.0000 , 0.8335]	[0.0000 , 0.4822]
$\mathcal{S}_\alpha(T(A^X = c))$	[0.5294 , 1.3020]	[0.7219 , 1.1025]	[0.7950, 1.0319]
$\mathcal{S}_\alpha(T(A^Y = e))$	[0.3659 , 1.3121]	[0.5294 , 1.0450]	[0.6098 , 0.9285]
$\mathcal{S}_\alpha(T(A^Y = d))$	[0.8113 , 1.4274]	[0.9248 , 1.2023]	[0.9580 , 1.1211]
$IG_\alpha(T, A^X)$	[0.1052 , 0.8902]	[0.4349 ,0.8419]	[0.5682 , 0.8188]
$IG_\alpha(T, A^Y)$	[-0.0863, 1.0434]	[0.2953 , 0.8975]	[0.4478 , 0.8282]
$IG^q_\alpha(T, A^X)$	[0.1299 , 0.8948]	[0.4155 , 0.8006]	[0.5379 , 0.7689]
$IG^q_\alpha(T, A^Y)$	[-0.2404 , 0.7549]	[-0.0088 , 0.5622]	[0.0935 , 0.4651]

aim to better recognize which attribute is the best to split on when there is too much uncertainty, with the goal to improve the overall accuracy of the model.

The query procedure mainly consists of two parts: ordering uncertain instances and asking for precise labels of those selected. Instances at a node are ordered by their impacts on the width of entropy interval. The one that influences the uncertain entropy interval most will be chosen for query. Denoting the instance set at a node as T, the optimal instance to be queried is selected as

$$I^{\alpha*} = \arg\max_i w(\mathcal{S}_\alpha(T)) - w(\mathcal{S}_\alpha(T \backslash < x_i, m_{y_i} >)) + \mathbf{1}_{m_{y_i}=vac} \qquad (21)$$

where $T \backslash < x_i, m_{y_i} >$ represents the instance set obtained by removing instance $< x_i, m_{y_i} >$ from T, and $\mathbf{1}_{m_{y_i}=vac}$ is the indicator function which has value 1 only when instance $< x_i, m_{y_i} >$ is vacuous.

The intuition behind Equation (21) is that if $< x_i, m_{y_i} >$ is vacuous, then $w(\mathcal{S}_\alpha(T)) = w(\mathcal{S}_\alpha(T \backslash < x_i, m_{y_i} >)$. To make sure that vacuous instances (whose querying have an important impact on the entropy interval) are queried often enough, we add 1 to (21) in their case. If $< x_i, m_{y_i} >$ is almost certain then most often we will have $w(\mathcal{S}_\alpha(T)) < w(\mathcal{S}_\alpha(T \backslash < x_i, m_{y_i} >))$, with

the difference being bigger as $< x_i, m_{y_i} >$ is more certain. The cases where $w(\mathcal{S}_\alpha(T)) > w(\mathcal{S}_\alpha(T \setminus < x_i, m_{y_i} >)$ concern those instances $< x_i, m_{y_i} >$ that are contradictory with the majority class, i.e., whose most plausible class is different from this latter.

In real applications, when the instance set T is of large scale, the impact of an instance on the entropy interval will become relatively small and there may be several instances having the same maximal value of (21). Therefore, to order query candidates, uncertainty of every instance is also considered. When optimal instances $I_1^{\alpha*}, \ldots, I_s^{\alpha*}$ have the same influence on $w(\mathcal{S}_\alpha(T))$, the one with larger uncertainty is more preferred, i.e., $I_p^{\alpha*} \succ I_q^{\alpha*}$ if $\sum_j pl_p(C_j) > \sum_j pl_q(C_j), p, q \in \{1, \ldots, s\}$.

Example 3. Consider the instances in Example 2, arranging the uncertain instances in node $A^X = a$, we achieve the query order shown in Table 3.

Table 3: Query order of node $A^X = a$

number	$w(\mathcal{S}_{0.9}(T)) - w(\mathcal{S}_{0.9}(T \setminus < x_i, m_i^Y >))$	$\sum_j pl_i(C_j)$
1	0.0806	1.4000
2	-0.0001	1.9000
4	-0.0001	1.3000
15	-0.0001	1.3000
13	-0.0330	1.4000
3	-0.0330	1.2000

From Table 1 we know that instance 1 is mislabelled and instance 2 is the most uncertain one. Our proposed query strategy does find out the most valuable instances.

Once query is requested at a decision node because we cannot decide what is the best split, we select n instances (usually $n = 1$) at every child node and label all those selected in one operation. Query is stopped when the number of queries reaches a fixed preset limit N_q. Denote $T_j^h = T(A^j = a_h^j), h = 1, \ldots, r_j$ the instance set within the child node corresponding to $A^j = a_h^j$, the query

strategy is summarized in Algorithm 2.

Algorithm 2: Algorithm of query strategy

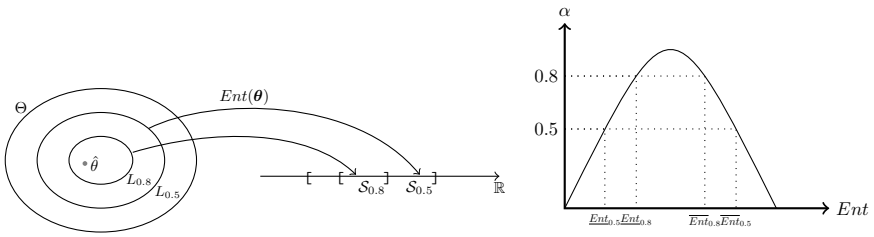
Input: attributes list $\mathscr{A}^q = \{A^1, ..., A^k\}$, instance set T, query limit N_q, number of queries num_q

Output: updated instance set T, number of queries num_q

```
T_q=findUncertain(T);
if |T_q|=0 then
    Return
for all A^j ∈ 𝒜^q and a_h^j ∈ A^j do
    I^{α*}_{T_j^h} = arg max_{x_i ∈ T_j^h} [w(S_α(T_j^h)) − w(S_α(T_j^h \ < x_i, m_{y_i} >))];
    if num_q < N_q then
        Return
    query the selected instance I^{α*}_{T_j^h};
    num_q = num_q + 1;
update T;
```

It can be noted that parameter α in the definition of α-cut can also control the number of queries to be made. As illustrated in Figure 3(a), as the value of α decreases, the α-cut L_α becomes larger, making the corresponding entropy interval $\mathcal{S}_\alpha$ wider or remaining the same. This can also be seen in the distribution shown in Figure 3(b) that a narrower entropy interval is achieved by a larger α, hence those intervals will often dominate each others, decreasing the number of queries needed to make a decision. Looking back to Example 2, entropy intervals achieved in the first splitting step under different α are shown in Table 2. As the value of α decreases, intervals become wider, making attribute selection more difficult.

Remark 1. When α equals to 1, the entropy interval degenerates to an exact entropy value computed by the optimal parameter $\hat{\boldsymbol{\theta}}$. This means that when all instances are precise and $\alpha = 1$, we retrieve the classical C4.5 method. $\alpha = 0$



(a) mappings under different α (b) distribution over α and entropy

Figure 3: Relations among α, α-cut L_α and entropy interval $\mathcal{S}_\alpha$

leads to a vacuous interval $[0, log_2(\ell)]$, whatever the number of instances is. This last behaviour is similar to what happens in imprecise probabilities when trying to learn from a completely vacuous prior [34, Sec 7.3.7]

3.4. Stopping criterion

After selecting the best splitting attribute, we split the decision node and attach the obtained structure to this node. This procedure is repeated recursively until meeting one of the following stopping criterion:

- No attribute available for selection;
- $\mid T \mid$ falls below a threshold;
- the estimated parameter $\hat{\boldsymbol{\theta}}$ for $\alpha = 1$ is unique and one component of it reaches 1;
- The upper bounds of information gain ratio intervals are all lower than zero.

When the stopping criterion is satisfied, the current node becomes a leaf node whose optimal parameter $\hat{\boldsymbol{\theta}}$ is the one maximizing the evidential likelihood ($\hat{\boldsymbol{\theta}}$ can be computed by using E^2M algorithm [23] or other convex optimization methods, as the contour function in this application has been proven to be concave). The leaf node is then labelled as the class corresponding to maximum component of $\hat{\boldsymbol{\theta}}$, i.e., labelled as $\hat{C}_\xi = \underset{C_i \in \mathscr{C}}{\arg} \max_i \hat{\theta}_i$, where $\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} L(\boldsymbol{\theta}; m_{\boldsymbol{y}})$.

The pseudo code of complete tree induction procedure is summarized in Algorithm 3. Once an active belief decision tree is built, the classification of a new instance can be realized by starting at the root node and going down according to its feature values until a leaf node is reached.

Algorithm 3: Induction of active belief decision trees (ABC4.5)

Input: uncertain training set T_{pl}

Output: final tree $Tree$

Create a root node containing all the instances in T_{pl};
if *stopping criterion is met* **then**
    $\hat{C}_\xi = \underset{C_i \in \mathscr{C}}{\arg} \max_i \hat{\theta}_i$;
    return $Tree$={root node};
else
    apply Algorithm 1 to select the splitting attribute A^{best};
    T_{pl}^v= induced subsets from T_{pl} based on A^{best};
    for *all* T_{pl}^v **do**
        $Tree^v = ABC4.5(T_{pl}^v)$;
        Attach $Tree^v$ to the corresponding branch of $Tree$;

Remark 2. The classical decision tree works as a special case of active belief decision tree where all instances are precise (all mass functions are both Bayesian and logical), $\alpha = 1$, and no query is carried out. The decision trees learned with evidential EM approach [21] (when only the output classes are uncertain) can also be retrieved from the proposed approach by setting α=1.

Example 4. We build the active belief decision tree for the 21 instances in Example 1 completely, meaning that we now check for each node $A^X = a$, $A^X = b$ and $A^X = c$ whether they can be split further. With parameters $\hat{\boldsymbol{\theta}}_{A^X=a} = (0, 0, 1)$, $\hat{\boldsymbol{\theta}}_{A^X=b} = (0, 1, 0)$ and $\hat{\boldsymbol{\theta}}_{A^X=c} = (0.3300, 0.6700, 0)$, nodes $A^X = a$, $A^X = b$ become leaves, and $A^X = c$ is split into two leaf nodes since $IG_\alpha(A^Y) = [0.5711, 1.0319]$ is positive. Figure 4 demonstrates the decision tree

learned from uncertain training data.

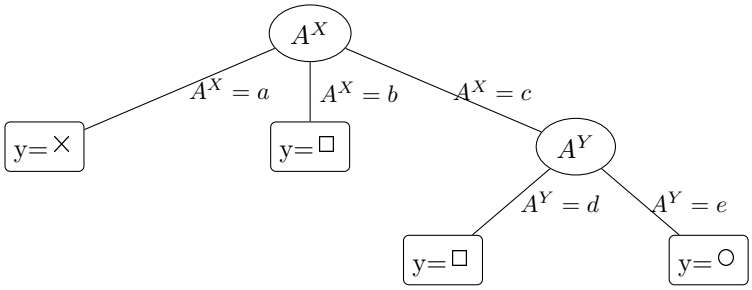


Figure 4: Active belief decision tree built for the illustrative example

4. Experiments on UCI data sets

In this section, we present experiments on various data sets and compare the classification performances of proposed active belief decision trees with respect to classical decision trees, E^2M decision trees and proposed belief decision trees without querying (using only (20) to select the best attribute). Since no uncertain data benchmark is available, we take data sets from the UCI repository [35] and introduce data uncertainty into them. 10-fold cross-validations on six UCI data sets are performed to validate our methodology. Details of these data sets are provided in Table 4.

Table 4: UCI data sets for validation

Data set	num of attributes	num of classes	num of instances
Breast tissue	9	6	106
Iris	4	3	150
Wine	13	3	178
Glass	9	7	214
Ecoli	8	7	336
Balance scale	4	3	625

Prior to tree induction, continuous attribute values are discretized by the method of equal width interval binning. Given an instance set

$$T = \begin{pmatrix} x_1^1, \cdots, x_1^k, y_1 \\ \vdots \\ x_n^1, \cdots, x_n^k, y_n \end{pmatrix} \text{ or } T = \begin{pmatrix} x_1^1, \cdots, x_1^k, m_{y_1} \\ \vdots \\ x_n^1, \cdots, x_n^k, m_{y_n} \end{pmatrix},$$

the range $[\underline{x^i}, \overline{x^i}]$ of each continuous-valued attribute $\boldsymbol{x^i} = (x_1^i, \cdots, x_n^i)^T$ is independently divided into four intervals of equal width by computing $\delta = \frac{\overline{x^i} - \underline{x^i}}{4}$ and setting thresholds at $\underline{x^i} + i\delta,\ i = 1, 2, 3$.

Given an uncertain observation m_{y_i}, denote C_i^* its true label. Thanks to the flexibility of belief functions, a high number of situations can be simulated from precise data:

- a *precise* observation is such that $pl_{y_i}(C_i^*) = 1$, and $pl_{y_i}(C_j) = 0, \forall C_j \neq C_i^*$
- a *vacuous* observation is such that $pl_{y_i}(C_j) = 1,\ \forall C_j \in \mathscr{C}$
- an *imprecise* observation is such that $pl_{y_i}(C_j) = 1$ if $C_j = C_i^*$ or $C_j \in \mathscr{C}_{rm}$, and $pl_{y_i}(C_j) = 0, C_j \in \mathscr{C} \setminus \{C_i^*,\ \mathscr{C}_{rm}\}$, where $\mathscr{C}_{rm}$ is a set of randomly selected labels
- an *uncertain* observation is such that $pl_{y_i}(C_i^*) = 1$, and $pl_{y_i}(C_j) = r_j, \forall C_j \neq C_i^*$, where r_j are sampled independently from uniform distribution $\mathcal{U}([0, 1])$
- a *noisy* observation is such that $pl_{y_i}(\tilde{C}) = 1$, and $pl_{y_i}(C_j) = 0, \forall C_j \neq \tilde{C}$, where the true label is replaced by a label $\tilde{C}$ uniformly drawn from $\mathscr{C}$

Since data generations are stochastic, all experiments are repeated five times to calculate the average result. In the following subsections, we set $N_q = n/4$ (a discussion of N_q can be seen in Section 4.3) and compare the performances of four trees:

- Tree 1 (classical $C4.5$), which only uses precise data in the training set during tree induction;

- Tree 2 (E^2M tree - $\alpha = 1$, without querying), which can deal with various kinds of uncertain data;
- Tree 3 ($ABC4.5$ - $\alpha = 0.8$, without querying), which takes both uncertain data and uncertain entropy into account;
- Tree 4 ($ABC4.5$ - $\alpha = 0.8$, with querying), which combines uncertain data, uncertain entropy and active learning to induce trees.

4.1. Experiments with vacuous data

Some instances in the training set are set to be totally vacuous (unobserved) while others remain precise. Vacuousness level $V \in [0, 1]$ controls the chance of an observation to become vacuous. For every observation, a number V_i is randomly generated on [0,1] and it will be replaced with a vacuous one if $V_i < V$.

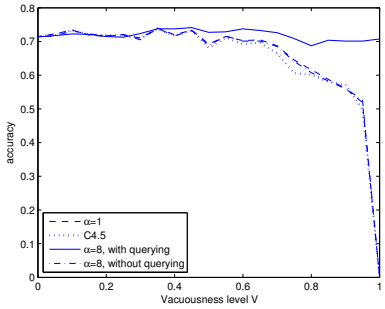
Algorithm 4: algorithm to generate vacuous observations from UCI data sets

Input: UCI data set $T = (\boldsymbol{x}, y)$, vacuousness level V

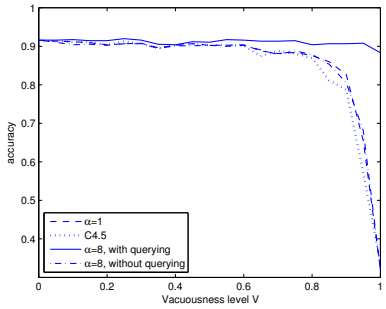
Output: training set $T_{pl} = (\boldsymbol{x}, pl)$

$\boldsymbol{x} \leftarrow discreteAttribute(\boldsymbol{x})$;
for $1 \leq i \leq n$ **do**
  $V_i \leftarrow randomGenerate(\mathcal{U}([0, 1]))$;
  if $V_i < V$ **then**
    $pl_{y_i}(C_j) = 1,\ \forall C_j \in \mathscr{C}$;
  else
    $pl_{y_i}(C_i^*) = 1,\ pl_{y_i}(C_j) = 0, \forall C_j \neq C_i^*$;

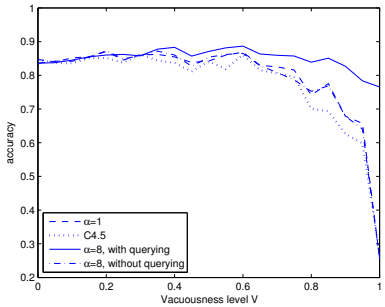
Given training sets generated by Algorithm 4, Figure 5 shows the classification accuracies changing with V. Accuracies of all the trees decrease as V increase, yet as V approaches 1, the curves show an obvious drop of accuracy except the one of active belief decision trees. Actually, with querying, we can



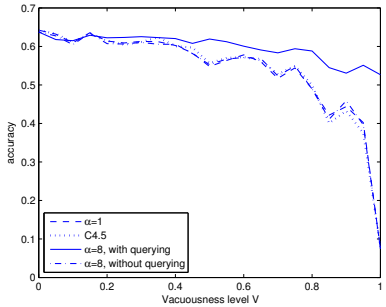
(a) Breast tissue data



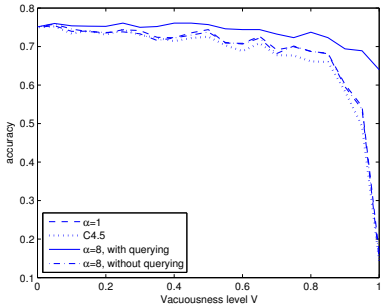
(b) Iris data



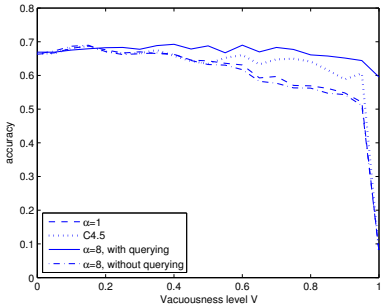
(c) Wine data



(d) Glass data



(e) Ecoli data



(f) Balance scale data

Figure 5: Experimental results with controlled vacuousness

get a satisfactory accuracy even if V reaches 1. On the other hand, when vacuousness is relatively low, our proposed approach can also guarantee similar performances as classical C4.5 trees, which makes active belief decision trees

suitable for all situations.

It can also be observed that active belief decision trees become more effective when data sets are complex: for simple data sets such as Iris data, normal methods only get bad results for a quite high degree of vacuousness ($V > 0.6$), while by querying vacuous instances, complex data sets see a clear accuracy improvement even when V is around 0.2, as seen in results of Ecoli data set.

4.2. Experiments with imprecise data

In this part, we consider the situation in which some data are imprecisely observed as set-valued, but the true value lies in the observed set (such observations are sometimes referenced as superset labels [36]). Imprecision level $I \in [0, 1]$ controls the percentage of imprecise observations. All instances in the training set are set initially to be precise. For every observation, a number I_i is uniformly generated on [0,1]. The observation is replaced with an imprecise one if $I_i < I$, otherwise, it remains precise.

To make an observation imprecise, generate a random number L_i ranging in [0,1] for every class label C_i (except the true label). Those labels with $L_i < I$ will have a plausibility equal to one. Note that $I = 1$ leads to a totally imprecise data set. Algorithm 5 summarizes the generation procedure.

As seen in Figure 6, active belief decision trees still have a quite stable performance with changing I. Accuracies of all four trees on all data sets decrease as imprecision increases. ABC4.5 reaches nearly the same performances as the other three trees in low-imprecision cases, and outperforms them when data are of high imprecision. The performances are quite similar as the results in previous subsection, yet as I grows, the proposed method may have a relatively smaller advantage over other methods than in experiments with controlled vacuousness. This can be partially explained by the fact that imprecise but non-totally vacuous instances brings less uncertainty to the data sets.

4.3. Experiments with noisy and uncertain data

To evaluate the performance of proposed approach in a more general situation, now we build plausibility distributions for every uncertain observation,

Algorithm 5: algorithm to generate imprecise observations from UCI data sets

Input: UCI data set $T = (\boldsymbol{x}, y)$, imprecision level I

Output: training set $T_{pl} = (\boldsymbol{x}, pl)$

$\boldsymbol{x} \leftarrow discreteAttribute(\boldsymbol{x})$;

for $1 \leq i \leq n$ **do**

  $pl_{y_i}(C_i^*) = 1$, $pl_{y_i}(C_j) = 0, \forall C_j \neq C_i^*$;

  $I_i \leftarrow randomGenerate(\mathcal{U}([0,1]))$;

  if $I_i < I$ **then**

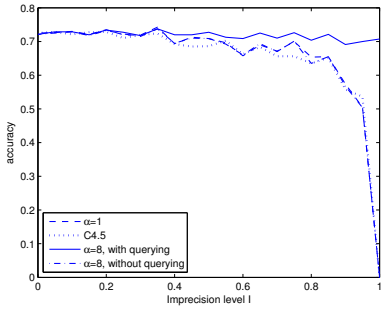
    $\mathscr{C}_{rm} \leftarrow randomGenerate(\mathscr{C} \setminus \{C_i^*\})$;

    $pl_{y_i}(\mathscr{C}_{rm}) = 1$;

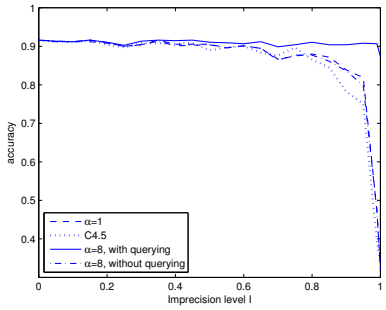
where the true label has a higher chance (but not necessarily) to have the highest plausibility.

Following procedure similar to those used previously, $N \in [0,1]$ decides whether we noise an observation by replacing its true label with another label uniformly drawn from $\mathcal{C}$. Uncertainty level $U \in [0,1]$ determines the chance of an observation to be uncertain. As depicted in Algorithm 6, we generate a random number $U_i \in [0,1]$ for each observation and set those having $U_i < U$ to be uncertain observations. The plausibility distribution for every uncertain observation is achieved by giving each label of that observation a random plausibility value ranging in $[0, U]$.

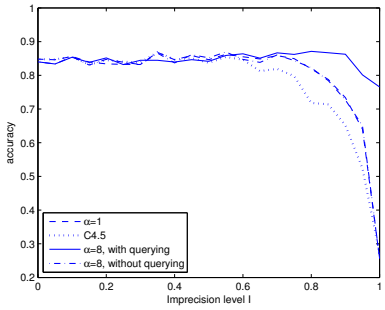
Fixing $N = 0.2$, the average classification accuracies changing with U are reported in Figure 7. Unlike classical C4.5 trees, the three trees based on evidential likelihood are pretty robust to data uncertainty. Thanks to the usage of evidential data and entropy intervals, they may even have a slight improvement of classification accuracies as U approaches 1. The reason for this behaviour is simple: when $U = 0$, the noisy data are all certain, hence their presence degrades in the same way the original C4.5 and the belief decision trees; when U increases, some of these noisy data (as well as non-noisy data) become un-



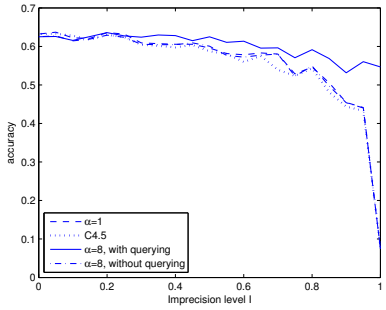
(a) Breast tissue data



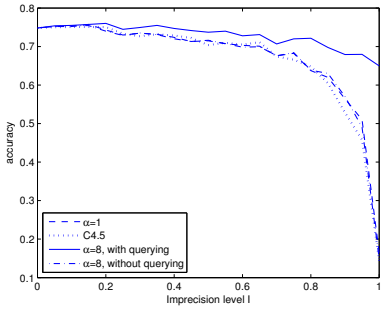
(b) Iris data



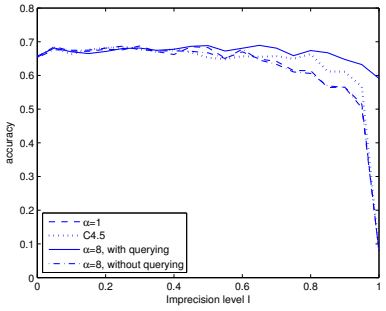
(c) Wine data



(d) Glass data



(e) Ecoli data



(f) Balance scale data

Figure 6: Experimental results with controlled imprecision

certain. When not allowing for querying, these uncertain noisy data will have a lower impact on the learning procedure, making it more robust. When allowing for querying, then some of these noisy information will be corrected into reli-

Algorithm 6: algorithm to generate noisy and uncertain observations

Input: UCI data set $T = (\boldsymbol{x}, y)$, noise level N, uncertainty level U

Output: training set $T_{pl} = (\boldsymbol{x}, pl)$

$\boldsymbol{x} \leftarrow discreteAttribute(\boldsymbol{x})$;

for $1 \leq i \leq n$ **do**

  $N_i \leftarrow randomGenerate(\mathcal{U}([0,1]))$;

  if $N_i < N$ **then**

    $C_i^* \leftarrow randomGenerate(\mathscr{C})$;

  $pl_{y_i}(C_i^*) = 1$, $pl_{y_i}(C_j) = 0, \forall C_j \neq C_i^*$;

  $U_i \leftarrow randomGenerate(\mathcal{U}([0,1]))$;

  if $U_i < U$ **then**

    $pl_{y_i}(C_j) \leftarrow randomGenerate(\mathcal{U}([0,U])),\ \forall C_j \neq C_i^*$;

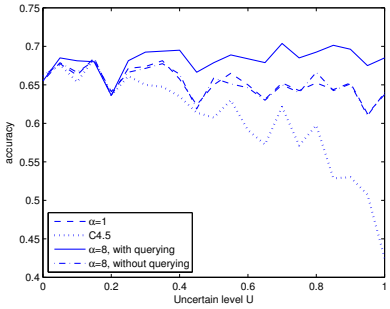
able precise information, therefore improving the quality of the induced model. This explains that allowing for uncertainty (or even injecting some) can actually improve the quality of the learned model when the training set contains some wrongly labelled instances.

With querying, the active belief decision trees can still result in the highest accuracy, but comparing to those without querying, the improvement will be relatively small. This can easily be explained by the fact that the true label remains the most plausible one, hence evidential likelihood methods performs quite well.

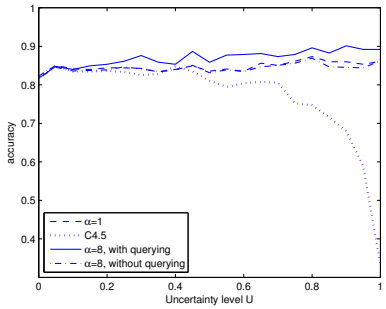
Similarly, Figure 8 reveals the classification behaviors with fixed $U = 0.5$ and changing N. From experimental results, it is obvious that active belief decision trees competes quite well for all the data sets. Also, the more noised the instances are, the greater the improvement can be.

4.4. Further experiments

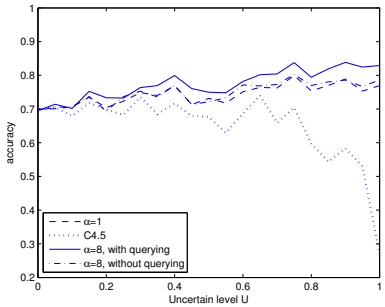
We can also look at the impacts of other control variables. For example, Figure 9 shows how accuracy involve when we set $U = 0.8$ and $N = 0.5$ (high



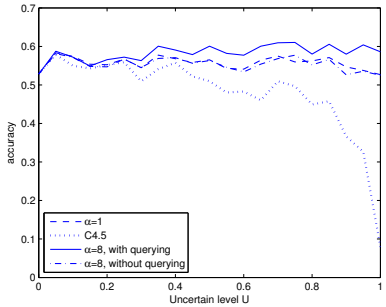
(a) Breast tissue data



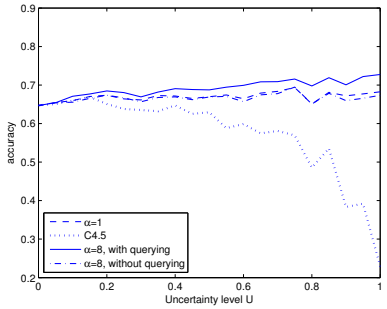
(b) Iris data



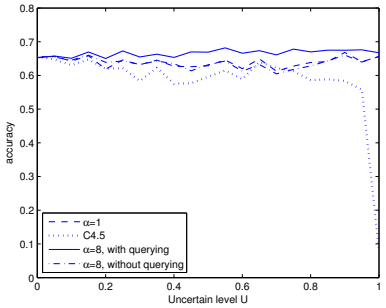
(c) Wine data



(d) Glass data



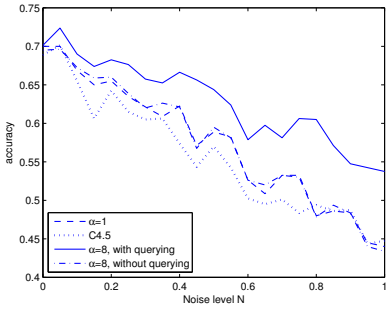
(e) Ecoli data



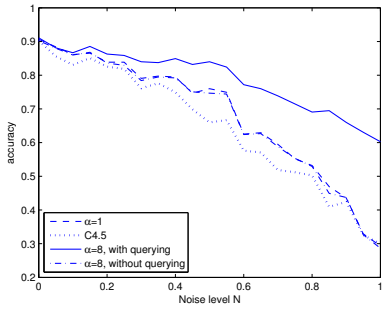
(f) Balance scale data

Figure 7: Experimental results with controlled uncertainty

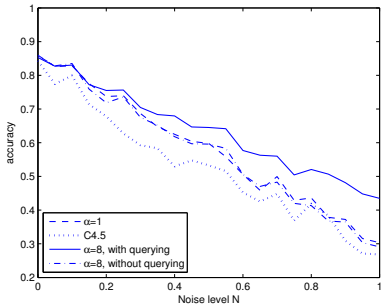
uncertainty and noise) and increase the number of queried items. As expected, we can see a steady increase in the accuracy with the number of queried items, with the trend slowing down as more data are queried. Most of the time, 60



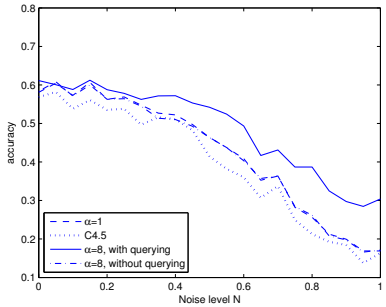
(a) Breast tissue data



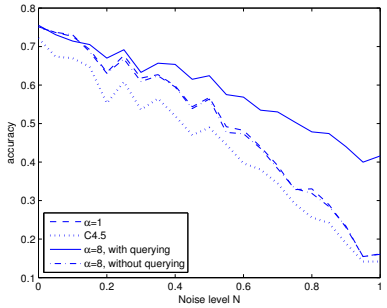
(b) Iris data



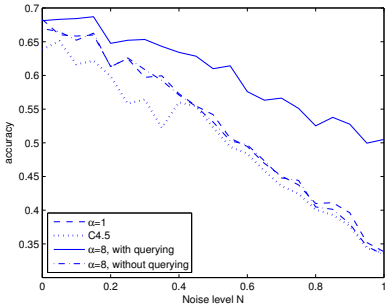
(c) Wine data



(d) Glass data



(e) Ecoli data

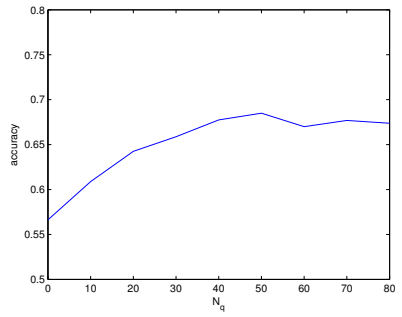


(f) Balance scale data

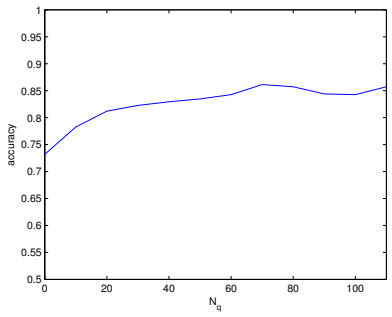
Figure 8: Experimental results with controlled noise

queries or less are sufficient to reach an accuracy close to the maximum (with the exception of the balance scale data set).

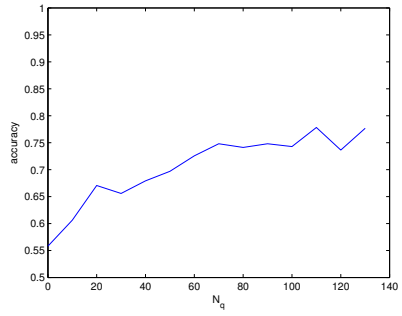
This indicates that querying data is worthy for different values of Nq (the



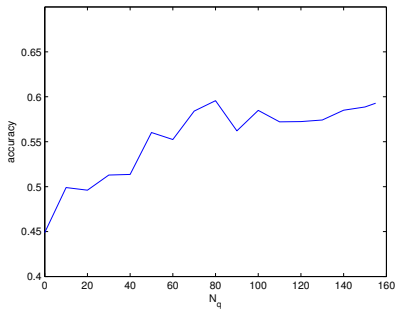
(a) Breast tissue data



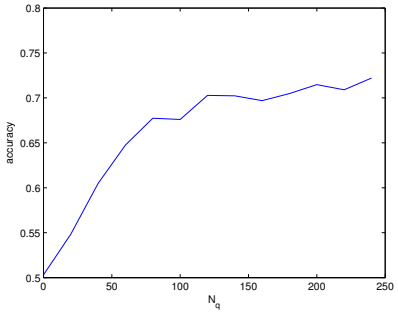
(b) Iris data



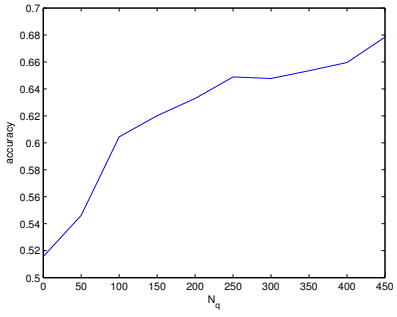
(c) Wine data



(d) Glass data



(e) Ecoli data



(f) Balance scale data

Figure 9: Experimental results with changing number of queries

number of queried items), even small ones, but that the more we can query, the better the model becomes.

5. Conclusion

Although much work has been done on learning trees from evidential data, they only focus on dealing with uncertainty rather than reducing this uncertainty through querying (active learning) to learn better models. In this paper, based on the generic model of belief functions, we have proposed an active belief decision tree learning approach which can improve classification accuracy by querying while learning. To deal with uncertain data, entropy intervals are extracted from the evidential likelihood to calculate gain ratio and select the best attribute. Meanwhile, to reduce uncertainty, we query the most valuable uncertain instances to help in the selection of the best attribute.

As the experimental results show, the proposed approach is robust to different kinds of uncertainties. Its performances are comparable to classical decision trees when data are precise, and can maintain those good performances even in extreme situations such as when data are very imprecise or noisy. Therefore, the active belief decision tree have a potentially broad field of application. Some improvements and related topics will be investigated in our future research, such as:

- To reduce the computational complexity to make it more efficient in large-scale data classification. Indeed, although the fact that the multinomial contour function is concave (Appendix Appendix A) allows for faster computations and reasonable query time for small data sets, it may become a problem for larger ones;
- To consider the uncertainty in both feature vectors and class labels, to achieve both feature and output querying. This would however require the development of efficient numerical estimations, as the contour function would then be non-concave;
- To study the pruning approach to reduce the size of decision tree and reduce over-fitting;

- To investigate different possible selection strategies for the instances to query, for instance taking inspiration from active learning criteria (e.g., expected model change).

References

[1] J. Josse, M. Chavent, B. Liquet, F. Husson, Handling missing values with regularized iterative multiple correspondence analysis, Journal of classification 29 (1) (2012) 91–116. doi:10.1007/s00357-012-9097-0.

[2] J. R. Quinlan, Decision trees as probabilistic classifiers, in: Proceedings of the Fourth International Workshop on Machine Learning, 1987, pp. 31–37. doi:10.1016/b978-0-934613-41-5.50007-6.

[3] S. Tsang, B. Kao, K. Y. Yip, W.-S. Ho, S. D. Lee, Decision trees for uncertain data, Knowledge and Data Engineering, IEEE Transactions on 23 (1) (2011) 64–78. doi:10.1109/tkde.2009.175.

[4] Y. Yuan, M. J. Shaw, Induction of fuzzy decision trees, Fuzzy Sets and systems 69 (2) (1995) 125–139. doi:10.1016/0165-0114(94)00229-z.

[5] X. Wang, X. Liu, W. Pedrycz, L. Zhang, Fuzzy rule based decision trees, Pattern Recognition 48 (1) (2015) 50–59. doi:10.1016/j.patcog.2014.08.001.

[6] B. Qin, Y. Xia, F. Li, Dtu: a decision tree for uncertain data, in: Advances in Knowledge Discovery and Data Mining, Springer, 2009, pp. 4–15.

[7] C. Liang, Y. Zhang, Q. Song, Decision tree for dynamic and uncertain data streams., in: ACML, 2010, pp. 209–224.

[8] C. C. Aggarwal, P. S. Yu, A survey of uncertain data algorithms and applications, IEEE Transactions on Knowledge and Data Engineering 21 (5) (2009) 609–623. doi:10.1109/tkde.2008.190.

[9] M. Mohri, Learning from uncertain data, in: Learning Theory and Kernel Machines, Springer, 2003, pp. 656–670.

[10] E. Huellermeier, Learning from imprecise and fuzzy observations: data disambiguation through generalized loss minimization, International Journal of Approximate Reasoning 55 (7) (2014) 1519–1534. doi:10.1016/j.ijar.2013.09.003.

[11] L. Liu, T. Dietterich, Learnability of the superset label learning problem, in: Proceedings of the 31st International Conference on Machine Learning (ICML-14), 2014, pp. 1629–1637.

[12] I. Couso, L. Sánchez, Harnessing the information contained in low-quality data sources, International Journal of Approximate Reasoning 55 (7) (2014) 1485–1486. doi:10.1016/j.ijar.2014.05.006.

[13] A. P. Dempster, Upper and lower probabilities induced by a multivalued mapping, The annals of mathematical statistics (1967) 325–339doi:10.1214/aoms/1177698950.

[14] G. Shafer, et al., A mathematical theory of evidence, Vol. 1, Princeton university press Princeton, 1976. doi:10.2307/1268172.

[15] C. Lian, S. Ruan, T. Denœux, An evidential classifier based on feature selection and two-step classification strategy, Pattern Recognitiondoi:10.1016/j.patcog.2015.01.019.

[16] Z.-g. Liu, Q. Pan, J. Dezert, G. Mercier, Credal classification rule for uncertain data based on belief functions, Pattern Recognition 47 (7) (2014) 2532–2541. doi:10.1016/j.patcog.2014.01.011.

[17] Z. Elouedi, K. Mellouli, P. Smets, Belief decision trees: theoretical foundations, International Journal of Approximate Reasoning 28 (2) (2001) 91–124. doi:10.1016/s0888-613x(01)00045-7.

[18] S. Trabelsi, Z. Elouedi, K. Mellouli, Pruning belief decision tree methods in averaging and conjunctive approaches, International Journal of Approximate Reasoning 46 (3) (2007) 568–595. doi:10.1016/j.ijar.2007.02.004.

[19] P. Vannoorenberghe, T. Denoeux, Handling uncertain labels in multiclass problems using belief decision trees, in: Proceedings of IPMU, Vol. 3, 2002, pp. 1919–1926.

[20] P. Vannoorenberghe, On aggregating belief decision trees, Information fusion 5 (3) (2004) 179–188. doi:10.1016/j.inffus.2004.01.001.

[21] N. Sutton-Charani, S. Destercke, T. Denœux, Learning decision trees from uncertain data with an evidential em approach, in: Machine Learning and Applications (ICMLA), 2013 12th International Conference on, Vol. 1, IEEE, 2013, pp. 111–116. doi:10.1109/icmla.2013.26.

[22] N. Sutton-Charani, S. Destercke, T. Denœux, Training and evaluating classifiers from evidential data: Application to e 2 m decision tree pruning, in: Belief Functions: Theory and Applications, Springer, 2014, pp. 87–94. doi:10.1007/978-3-319-11191-9_10.

[23] T. Denœux, Maximum likelihood from evidential data: an extension of the em algorithm, in: Combining Soft Computing and Statistical Methods in Data Analysis, Springer, 2010, pp. 181–188. doi:10.1007/978-3-642-14746-3_23.

[24] B. Settles, Active learning literature survey, University of Wisconsin, Madison 52 (55-66) (2010) 11.

[25] A. Bordes, S. Ertekin, J. Weston, L. Bottou, Fast kernel classifiers with online and active learning, The Journal of Machine Learning Research 6 (2005) 1579–1619.

[26] T. Denoeux, Likelihood-based belief function: justification and some extensions to low-quality data, International Journal of Approximate Reasoning 55 (7) (2014) 1535–1547. doi:10.1016/j.ijar.2013.06.007.

[27] J. R. Quinlan, Induction of decision trees, Machine learning 1 (1) (1986) 81–106. doi:10.1007/bf00116251.

[28] S. B. Kotsiantis, Decision trees: a recent overview, Artificial Intelligence Review 39 (4) (2013) 261–283.

[29] J. R. Quinlan, C4. 5: programs for machine learning, Elsevier, 2014. doi:10.1007/bf00993309.

[30] L. Breiman, J. Friedman, C. J. Stone, R. A. Olshen, Classification and regression trees, CRC press, 1984.

[31] T. Denoeux, Maximum likelihood estimation from uncertain data in the belief function framework, Knowledge and Data Engineering, IEEE Transactions on 25 (1) (2013) 119–130. doi:10.1109/tkde.2011.201.

[32] A. Birnbaum, On the foundations of statistical inference, Journal of the American Statistical Association 57 (298) (1962) 269–306.

[33] T. Reineking, K. Schill, Evidential object recognition based on information gain maximization, in: Belief Functions: Theory and Applications, Springer, 2014, pp. 227–236.

[34] P. Walley, Statistical reasoning with imprecise probabilities, Monographs on statistics and applied probability, Chapman and Hall, 1991.
URL https://books.google.fr/books?id=-hbvAAAAMAAJ

[35] M. Lichman, UCI machine learning repository (2013).
URL http://archive.ics.uci.edu/ml

[36] L. Liu, T. G. Dietterich, A conditional multinomial mixture model for superset label learning, in: Advances in neural information processing systems, 2012, pp. 557–565.

Appendix A. Concavity of multinomial evidential likelihood

Contour function $pl_\Theta(\boldsymbol{\theta}; m_{\boldsymbol{y}})$ is a concave function.

$$pl_\Theta(\boldsymbol{\theta}; m_{\boldsymbol{y}}) = \frac{\mathbb{E}_{\boldsymbol{\theta}}[pl_y(Y)]}{\sup_{\boldsymbol{\theta}\in\Theta} \mathbb{E}_{\boldsymbol{\theta}}[pl_y(Y)]} = \frac{\prod_{i=1}^n \sum_{j=1}^\ell \theta_j pl_i(j)}{\sup_{\boldsymbol{\theta}\in\Theta} \prod_{i=1}^n \sum_{j=1}^\ell \theta_j pl_i(j)}$$

Proof. (i) The domain of function pl_Θ

$$\Theta = \{\boldsymbol{\theta} | \theta_i \in [0,1], \sum_i \theta_i = 1, i = 1, ..., \ell\}$$

is a unit simplex, hence a convex set.

(ii) Given observations $\boldsymbol{y}$, take two vectors $\boldsymbol{\theta}$ and $\boldsymbol{\theta'}$ from Θ, define

$$g(\boldsymbol{\theta}) = \log(L(\boldsymbol{\theta}; m_{\boldsymbol{y}})) = \sum_{i=1}^n \log \sum_{j=1}^\ell \theta_j pl_i(j).$$

For any $\lambda \in [0,1]$, we have

$$g(\lambda\boldsymbol{\theta} + (1-\lambda)\boldsymbol{\theta'}) = \sum_{i=1}^n \log \sum_{j=1}^\ell (\lambda\theta_j + (1-\lambda)\theta'_j) pl_i(j)$$

and

$$\begin{aligned}
&\lambda g(\boldsymbol{\theta}) + (1-\lambda) g(\boldsymbol{\theta'}) \\
&= \sum_{i=1}^n [\lambda \log \sum_{j=1}^\ell \theta_j pl_i(j) + (1-\lambda) \log \sum_{j=1}^\ell \theta'_j pl_i(j)] \\
&= \sum_{i=1}^n \log[(\sum_{j=1}^\ell \theta_j pl_i(j))^\lambda \cdot (\sum_{j=1}^\ell \theta'_j pl_i(j))^{1-\lambda}].
\end{aligned}$$

Then,

$$\begin{aligned}
&g(\lambda\boldsymbol{\theta} + (1-\lambda)\boldsymbol{\theta'}) - [\lambda g(\boldsymbol{\theta}) + (1-\lambda) g(\boldsymbol{\theta'})] \\
&= \sum_{i=1}^n \log \frac{\sum_{j=1}^\ell [\lambda\theta_j + (1-\lambda)\theta'_j] pl_i(j)}{[\sum_{j=1}^\ell \theta_j pl_i(j)]^\lambda \cdot [\sum_{j=1}^\ell \theta'_j pl_i(j)]^{1-\lambda}} \\
&= \sum_{i=1}^n \log \frac{\lambda \sum_{j=1}^l \theta_j pl_i(j) + (1-\lambda)[\sum_{j=1}^\ell \theta'_j pl_i(j)]}{[\sum_{j=1}^\ell \theta_j pl_i(j)]^\lambda \cdot [\sum_{j=1}^\ell \theta'_j pl_i(j)]^{1-\lambda}}
\end{aligned}$$

According to the weighted arithmetic mean and weighted geometric mean inequality, for any $y_1, y_2, \lambda \geq 0, \lambda y_1 + (1-\lambda) y_2 \geq y_1^{\lambda} \cdot y_2^{1-\lambda}$, therefore

$$g(\lambda \boldsymbol{\theta} + (1-\lambda)\boldsymbol{\theta}') - [\lambda g(\boldsymbol{\theta}) + (1-\lambda) g(\boldsymbol{\theta}')] \geq 0$$

$g(\boldsymbol{\theta})$ is a concave function. Since exponential function is convex and increasing, the composition $e^{g(\boldsymbol{\theta})}$ remains concave. Normalization will not change concavity as well, hence pl_{Θ} is a concave function. □