Self-Learning from Pairwise Credal Labels

Vitor Martin Bordini, Sébastien Destercke, Benjamin Quost

Self-Learning from Pairwise Credal Labels

Vitor Martin Bordini[*], Sebastien Destercke and Benjamin Quost

Université de technologie de Compiègne, CNRS, Alliance Sorbonne Université, Heudiasyc, Compiègne, France

Abstract. Self-learning is a popular machine learning approach to compensate for the lack of labeled data, in instances associated with "pseudo-labels" produced by the trained model are used. Such an approach can degrade the solution when the provided pseudo-labels are of a scarce quality. A solution is to take into account the labeling uncertainty. We advocate the use of rich uncertainty models, which make it possible to better account for the lack of information or imprecision attached with pseudo labels. We address the specific case where the model is obtained via a one-versus-rest decomposition of the set of classes.

1 Introduction

In machine learning, a critical issue for training a classification model is to find sufficient data with high-quality labels. Producing labels for data is often expensive because it requires time and in some cases specialized work. To minimize the dependence to such labels, semi-supervised learning aims at making use of both labeled and unlabeled data at hand to train the model.

Although this paradigm is far from new, it has recently gained attention with the popularization of algorithms such as FixMatch [11]. This algorithm uses a specific strategy called self-supervised learning, where the model replaces missing labels with its own predictions, called *pseudo-labels*. Algorithms such as FixMatch [11] frequently rely on data augmentation and pseudo-label selection to improve the performances of the model.

Obviously, this strategy may induce a bias when re-training the model, even if a selection step is employed. Since pseudo-labels can be viewed as degenerate probability distributions (putting a unit probability mass on a single class), Lienen *et al.* [7] proposed to replace them with convex probability sets of a specific kind, i.e., possibility distributions. Possibility distributions combines the advantages of being more expressive with simplicity, but still have the downside that their corresponding probability sets include at least a degenerated one.

More recently, Côme *et al.* [10] generated credal sets using Inductive Venn-Abers Prediction (IVAP), an approach which produces probability intervals [4]. However, Venn-Abers Predictors are limited to binary problems. A way to bypass this issue was presented by Manokhin [8], who generalized IVAP to the multi-class case, essentially by transforming a multi-class problem via a pairwise decomposition. Inspired by this work, we propose a self-learning procedure that uses IVAP in a multi-class setting by using a pairwise approach.

The paper is structured as follows. We present our approach as well as the required elements to understand it in Section 2. Section 3 than presents experiments on various data sets, together with a discussion of their results. Section 4 concludes the paper and provides several future possible research directions.

[*] Corresponding Author. Email: vitor.martin-bordini@hds.utc.fr

2 Proposed approach

2.1 Reminders on standard learning

We start by recalling the basics of standard semi-supervised learning. Assume an input space $\mathcal{X}$ and an output space $\mathcal{Y} = \{y^1, y^2, ..., y^\lambda\}$ with a probability space $P(\mathcal{Y})$, where λ is the number of classes. Paired samples of $\mathcal{X}$ and $\mathcal{Y}$, as well as samples of $\mathcal{X}$ only, constitute the dataset $\mathcal{D} = \mathcal{D}_L \cup \mathcal{D}_U$, where $\mathcal{D}_L = \{z_1 = (x_1, p_1), z_2 = (x_2, p_2), ..., z_n = (x_n, p_n)\}$ and $\mathcal{D}_U = \{(x_{n+1}), ..., (x_{n+m})\}, x \in \mathcal{X}, p \in \mathcal{P}(\mathcal{Y})$. Plainly put, $\mathcal{D}_L$ is composed from n labeled samples and $\mathcal{D}_U$ of m unlabeled samples, with the usual assumption that $m >> n$; $\mathcal{D}_L$ and $\mathcal{D}_U$ are assumed to be i.i.d., and independent from each other. For simplification, we denote $p_j^i = P(Y = y_i | X = x_j)$ and $p^i = P(Y = y_i)$. Note that $p_1, \ldots, p_n$ are degenerate, i.e., if y_i^k is the label of x_i, $p_i^j = 1$ for $j = k$, zero else.

The problem of learning a probabilistic classifier is then to estimate a predictive function $h_\theta : \mathcal{X} \times \Theta \to P(\mathcal{Y})$. Should we have only $\mathcal{D}_L$ at our disposal, this is classically done by finding an optimal empirical risk minimiser

$$\theta^* = \arg\min_\theta R = \arg\min_\theta \sum_{(x_i, p_i) \in \mathcal{D}_L} \mathcal{L}(p_i, h_\theta(x)), \quad (1)$$

where $\mathcal{L} : [0,1] \times [0,1] \to \mathcal{R}$ is a loss function. In this paper, we will adopt the Kullback-Leibler divergence

$$\mathcal{L}(p, h(x_j)) := \mathcal{D}_{KL}(p \| \hat{p}) = \sum_{i=1}^{\lambda} p_j^i \log \frac{p_j^i}{\hat{p}^i(x_j)} \quad (2)$$

as the loss function, with $\hat{p}^i(x_j)$ the estimated probability derived from h. Semi-supervised learning then aims at using the information in $\mathcal{D}_U$ in different ways. In this paper, we will explore one of these ways, where the model h_θ is iteratively used to pseudo-label the instances and re-trained using the resulting set of (pseudo-)labeled data.

The classifier should return predictions that are both reliable and informative. Therefore, it seems desirable to consider

pseudo-labels together with a quantification of their uncertainty rather than hard pseudo-labels with only the appearance of certainty: the former will arguably bias the model way less than the latter, mitigating the impact of erroneous predictions and thus converging to a better model. In this paper, we extend an approach devoted to binary classification [10] to the multi-class setting, where we consider sets of probabilities (or credal sets) obtained by a calibration step as pseudo-labels.

2.2 Learning from Credal Sets

It has been argued by many authors that probabilities may be too limited to account for all facets of uncertainty, and in particular those arising from a lack of knowledge [14]. In this paper, we therefore propose to replace a precise probability distribution p with a credal set $K \subseteq P(\mathcal{Y})$. However, classical loss functions $\mathcal{L} : P(\mathcal{Y}) \times P(\mathcal{Y}) \to \mathcal{R})$ cannot be applied any more, since credal sets are convex subsets of $P(\mathcal{Y})$ and not elements thereof. Consequently, it is imperative to adapt the definition of loss function so that pseudo-labels can be used.

In this paper, we use an optimistic approach, which consists in selecting the minimum value $\mathcal{L}_{min}$ of a classical loss function inside the credal set:

$$\mathcal{L}_{min}(K, h(x)) = \min_{p \in K} \mathcal{L}(p, h(x)). \tag{3}$$

This approach has been largely studied for semi-supervised learning [1, 9, 2], and has been shown to exhibit good performances in a co-learning setting [12], which is similar to the self-training setting considered here.

Our learning procedure consists in computing the optimal parameter θ^* of the model h by plugging-in the minimal loss (3) into Equation (1) and solving the min-min problem

$$\theta^*_{opt} = \arg\min_{\theta} R_{opt} := \sum_{(x_i, K_i) \in \mathcal{D}_L} \mathcal{L}_{min}(K_i, h_\theta(x)), \tag{4}$$

where we assume that each label can now be a credal set—note that this encompasses all previous settings, with degenerate or precise probability distributions being special cases of credal sets.

2.3 Precise one-versus-all decomposition

To extend the binary case mentioned above to a multi-class setting, we consider the "one-vs-all" approach: each of the classes y^k is in turn opposed to all the others in a binary classification problem. A direct consequence is that we handle a set of classifiers $\{h^k_\theta : \mathcal{X} \times \Theta \to P(\mathcal{B}^k), k = 1, \ldots, \lambda\}$, with $\theta \in \Theta$ the parameters of the model and $\mathcal{B}^k = \{y_k, \neg y_k\}$. Since we have λ classes, we retrieve λ classifiers, each trained over a binary (and hopefully easier) classification problem. This approach is appealing since the probability estimated by h^k_θ is the same as a probability estimated for class y_k in the multi-class problem, i.e, $P(y^k|\mathcal{B}^k)$ can be identified with $P(y^k)$. For the sake of simplicity, we will write $h^k_\theta(x) := \hat{p}(y^k|x)$.

In the precise case, each classifier h^k_θ is trained by finding the optimal parameter θ^*_k minimizing the empirical risk R^k:

$$\theta^*_k = \arg\min_{\theta} R^k(p^k, h^k_\theta) = \arg\min_{\theta} \sum_{(x, p^k) \in \mathcal{D}^k_L} \mathcal{L}(p^k, h^k_\theta(x)). \tag{5}$$

Note that the estimates $h^k_\theta(x)$ may not be consistent across all classifiers, i.e., we may have $\sum_{y^k} h^k_\theta(x) \neq 1$: then, a simple way to make them consistent is to take the normalized values as final estimates, that is $\hat{p}(y^k|x) = {}^{h^k_\theta(x)}/_{\sum_{y^k} h^k_\theta(x)}$. Figure 1 presents a scheme of the model.

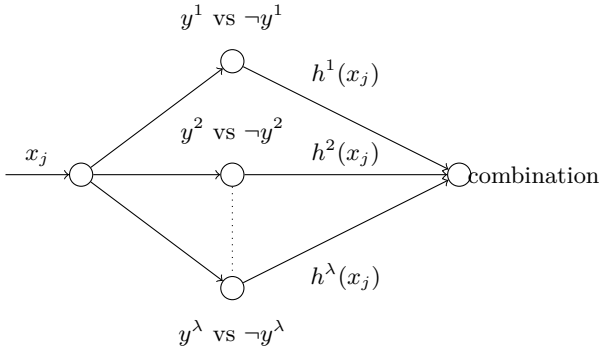


Figure 1: Prediction for output x_j

2.4 Credal one-versus-all decomposition

The credal setting amounts to replace precise estimates p^k by their imprecise counterparts $[\alpha^k, \beta^k]$, as illustrated in Figure 2. Before explaining the IVAP part of this graph in Section 2.5, we focus on the interval and combination parts.

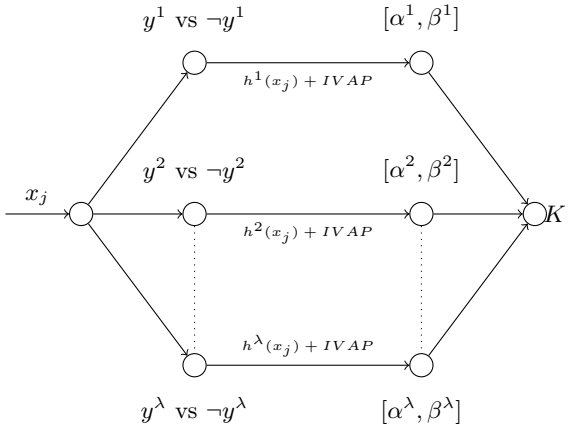


Figure 2: Credal Learning

The set of classifiers issued from the one-vs-all decomposition generates the credal set

$$K = \{p \in P(\mathcal{Y}) | \forall k = 1, \ldots, \lambda, \alpha^k \leq p(y_k) \leq \beta^k\}. \tag{6}$$

Computing Equation (3) in the case of the KL divergence (2) then amounts to solve the problem

$$\begin{aligned} \min_{p} \quad & \sum_{k=1}^{\lambda} p^k_j \log {}^{p^k_j}/_{\hat{p}^k(x_j)} \\ \text{s.t.} \quad & p^k_j \leq \beta^k_j, \quad p^k_j \geq \alpha^k_j, \quad \sum_{k=1}^{\lambda} p^k = 1, \end{aligned} \tag{7}$$

since α^k (resp., β^k) is a lower (resp., upper) bound for $p_k := p(y_k)$. Note that K is not guaranteed to be non-empty: if the credal set is improper (i.e, the condition $\sum \alpha^k \leq 1 \leq \sum \beta^k$ is not satisfied [4]), the problem (7) does not have any solution.

To overcome this issue, we may discount the credal set K, by weakening the lower or upper bounds in order to enlarge it to $K' = \{p \in P(\mathcal{Y}) | \forall k, \alpha'^k \leq p^k \leq \beta'^k\}$, such that $K \subseteq K'$. It is desirable that K' remains tight: the conditions $\sum_{i \neq j} \alpha'^i + \beta'^j \leq 1$ and $\sum_{i \neq j} \beta'^i + \alpha'^j \geq 1$ should be satisfied, while changing K as little as possible. In this work, two cases are considered for the discounting.

- Improper by lower bounds: whenever $\sum \alpha^i \geq 1$, the reason for not finding solution to Eq. (7) is that α is too big: then, we diminish α by a discounting factor δ such that $\sum_{i=1}^{\lambda} \alpha^i - \delta^i = \alpha'^i \leq 1$. We also need lower bounds to be at least 0, thus $0 \leq \alpha^i - \delta^i \leq 1 \rightarrow 0 \leq \delta^j \leq \alpha^j$.

$$\begin{aligned} &\min_{\delta} \sum \delta^i \\ \text{s.t.} \quad &\sum_{i \neq j} (\alpha^i - \delta^i) + \beta^j \leq 1 \\ &0 \leq \delta^j \leq \alpha^j \end{aligned} \tag{8}$$

- Improper by upper bounds: whenever $\sum \beta^i \leq 1$, the reason for not finding solution to Eq. (7) is that β is too small. Consequently, we increase β by a discounting factor ϵ such that $\sum_{i=1}^{\lambda} \beta^i + \epsilon^i = \beta'^i \geq 1$. We also need upper bounds to be smaller than 1, thus $\beta^i \leq \beta^i + \epsilon^i \leq 1 \rightarrow 0 \leq \epsilon^i \leq 1 - \beta^i$.

$$\begin{aligned} &\min_{\epsilon} \sum \epsilon^i \\ \text{s.t.} \quad &\sum_{i \neq j} (\beta^i + \epsilon^i) + \alpha^j \geq 1 \\ &0 \leq \epsilon^j \leq 1 - \beta^j \end{aligned} \tag{9}$$

Example 1. *Consider a three-class problem where our imprecise model has predicted the set K such as*

$$\begin{aligned} p^1 &\in [\alpha^1 = 0, \beta^1 = 0.2], \\ p^2 &\in [\alpha^2 = 0.6, \beta^2 = 0.7], \\ p^3 &\in [\alpha^3 = 0.5, \beta^3 = 0.55]. \end{aligned}$$

We can see that K is improper since $\sum_{k=1}^{3} \alpha^k = 1.1 \geq 1$. We relax the constraints so that a possible solution can be found, while altering K as few as possible, by solving (8)*. The solution of this problem is*

$$\delta = \begin{bmatrix} 0.0 \\ 0.15 \\ 0.2 \end{bmatrix} \tag{10}$$

Consequently, the discounted credal set K' becomes

$$\begin{aligned} p^1 &\in [\alpha^1 - \delta^1 = 0, \beta^1 = 0.2], \\ p^2 &\in [\alpha^2 - \delta^2 = 0.45, \beta^2 = 0.7], \\ p^3 &\in [\alpha^3 - \delta^3 = 0.4, \beta^3 = 0.55]. \end{aligned}$$

■

2.5 Inductive Venn-Abers Predictors

The intervals $[\alpha^k, \beta^k]$ used in the previous section are generated via Inductive Venn-Abers Prediction (IVAP) [13] for each binary classifier. IVAP is a calibration method suited to binary problems. It is a special case to Inductive Venn Predictors, with the advantage to be computationally efficient and simple to implement.

IVAP generates calibrated intervals $[\alpha^k, \beta^k]$, interpreted as lower/upper bounds on the probability $P(Y = y_k | \mathcal{B}^k)$, based on the predictions of the binary classifier $h^k(x)$. IVAP relies on the fact that many classifiers produce scores s as outputs, which can be compared to a threshold c to make a decision—e.g., we select for instance class y_1 iff $s \geq c$. The scores may be "calibrated" based on a non-decreasing function g. IVAP basically determines this function by fitting an isotonic regressor to the calibration set and the instance to be classified (see Algorithm 1).

Algorithm 1 Inductive Venn-Abers Predictors

Require: Calibration set $\mathcal{D}_C = \{(x_1, p_1), ..., (x_n q, p_q)\}$;
Require: j^{th} point to predict x_i.
Require: Binary classifier h^i,i.e, $h^i(x) = \hat{P}(Y = y_i | \mathcal{B}^i)$.
Find scores on the set $\{x_1, .., x_q, x_j\}$,i.e, calculate the set $\{s_1 = h^i(x_1), ..., s_q = h^i(x_q), s_j = h^i(x_j)\}$
For each possible class (0 or 1) fit two isotonic regressors g_0 and g_1 using sets $\{(s_1, p_1), ..., (s_q, p_q), (s_j, 0)\}$ and $\{(s_1, p_1), ..., (s_q, p_q), (s_j, 1)\}$
Return $[p_0, p_1] := [g_0(s_j), g_1(s_j)]$

In practice, we use IVAP to build up the intervals in Figure 2. Note that Algorithm 1 requires a calibration data set $\mathcal{D}_C$ of labeled data, usually sampled from the training set.

2.6 Proposed approach

We now describe our self-supervised credal learning approach in a multi-class setting based on probability intervals. It initially requires three data sets: a labeled data set $\mathcal{D}_L$, typically of limited size, to learn the first instances of the model; a labeled calibration data set $\mathcal{D}_C$ used in Algorithm 1 to produce credal self-supervised labels; and an unlabeled data set $\mathcal{D}_U$, progressively (partially) completed with credal labels. The procedure is summarized in Algorithm 2.

Note that Algorithm 2 solves the global optimization problem 7 once, and then uses the obtained probability distribution as a soft label then fed to every pairwise classifier–pending on some class receiving a high enough probability mass. The probability distribution is obtained in Line 19, while Line 20 builds the set of retained instances for the current loop, and the ensuing loop creates the corresponding binary labels. Finally, we end the loop whenever we cannot add instances with high enough probabilities to the data set.

3 Experiments

3.1 Strategies

For each binary problem, we trained a neural network with a hidden layer of m neurons with learning rate τ (depending on the dataset) on $\mathcal{D}_T^i$, and then applied our method over 50 iterations for all datasets. We chose a neural network since it is often poorly calibrated [6]. The optimizer was SGD with no momentum nor weight decay. Batch size was set to 10. We split each dataset into four new sets: $\mathcal{D}_L$ with 15% of the

Algorithm 2 Credal labeling strategy

Require: Labeled set $\mathcal{D}_L = \{(x_1, p_1), ..., (x_n, p_n)\}$, Unlabeled set $\mathcal{D}_U = \{(x_{n+1}), ..., (x_{n+m})\}$, Calibration data set $\mathcal{D}_C$;
Require: Threshold δ;
for i from 1 to λ **do**
  Initialize model h^i;
  Build set $\mathcal{D}_L^i = \{(x_1, p_1^i), ..., (x_n, p_n^i)\}$;
  Fit model h^i on $\mathcal{D}_L^i$;
end for
repeat
  for i from 1 to λ **do**
    $\hat{p}_k^i = h^i(x_k)$;
    Using Algorithm 1 with $\mathcal{D}_C$ to get $[\alpha_k^i, \beta_k^i]$;
  end for
  Build set $K_{x_k} = \{p \in P(\mathcal{Y}) | \forall i \in [1, \lambda]; \alpha_k^i \leq p_k^i \leq \beta_k^i\}$;
  Check if K_{x_k} is proper, i.e, $\sum_i \alpha^i \leq 1 \leq \sum_i \beta^i$;
  if $\sum_i \alpha^i \geq 1$ **then**
    Solve problem (8);
  end if
  if $\sum_i \beta^i \leq 1$ **then**
    Solve problem (9);
  end if
  Solve problem (7) to obtain $\tilde{p_k}$;
  Build set $\mathcal{D} = \{x_k \in \mathcal{D}_U | \exists j \text{ s.t. } \tilde{p}_k^j \geq \delta\}$;
  for i from 1 to λ **do**
    Build set $\mathcal{D}_U^i = \{(x_k, \tilde{p}_k^i) | x_k \in \mathcal{D}\}$;
    $\mathcal{D}_T^i = \mathcal{D}_L^i \cup \mathcal{D}_U^i$;
    Fit model h^i on $\mathcal{D}_T^i$;
  end for
  $\mathcal{D}_U = \mathcal{D}_U \setminus \mathcal{D}$;
until $\mathcal{D}_U$ is not empty

samples, $\mathcal{D}_U$ with 65% of the samples, $\mathcal{D}_C$ with 7 samples per class (Iris and Wine only have 2 samples per class because they are too small) and $\mathcal{D}_t$ with 20% of the samples. We compare three different strategies:

- Hard labels: a standard, classical self-learning (SL) procedure consisting of adding a batch of new labeled data at each iteration (the batch of data for which the prediction probabilities are the highest). That strategy is similar to the one used by Fixmatch [11]. However, in our case, we don't use data augmentation (contrary to FixMatch) because it is not well defined for tabular datasets. The algorithm of this strategy is described in Algorithm 3.
- Credal Labels: We start by learning a first classifier h_{θ^0} on the fully labeled set $\mathcal{D}_T$, through standard loss minimization (Eq. (1)). We then apply IVAP on each binary problem to produce intervals that are combined to generate credal labels on the instances in $\mathcal{D}_U$. We denote by $K_{x_k}^0$ the credal set $\{p | \forall i = 1, 2, ..., \lambda; \alpha_k^i \leq p_k^i \leq \beta_k^i\}$ obtained for observation x_j. If K is not proper, we solve problems (8) or (9). For all x_k we solve problem (7) and obtain $\tilde{p}_k$. We thus rebuild $\mathcal{D}_U$ into set $\{(x_{n+1}, \tilde{p}_{n+1}), ..., (x_{n+m}, \tilde{p}_{n+m})\}$. Samples whose $\max \tilde{p}$ are bigger than a certain threshold δ are selected on the set $\mathcal{D}_U^0$, which is then added to the labeled set: $\mathcal{D}_L^0 \leftarrow \mathcal{D}_L \cup \mathcal{D}_U^0$. We then fit our model into $\mathcal{D}_L^0$ to obtain a new model h_{θ^1}, and so on. This iterative procedure is repeated until until no new unlabeled sample can be added to the labeled set (see Algorithm 2).
- Soft labels: at each iteration k, we label $\mathcal{D}_U$ with $h_{\theta^k}(x)$. Pseudo-labels that are bigger than a threshold δ are selected and added into the labeled set, giving the new set $\mathcal{D}_L^k \leftarrow \mathcal{D}_L \cup \mathcal{D}_U^k$. We then fit our model on $\mathcal{D}_L^k$. We retrain the model using this new set until all unlabeled samples are added to the labeled set. The procedure is presented in Algorithm 4.

Algorithm 3 Hard label strategy

Require: Labeled set $\mathcal{D}_L = \{(x_1, p_1), ..., (x_n, p_n)\}$, Unlabeled set $\mathcal{D}_U = \{(x_{n+1},), ..., (x_{n+m},)\}$;
Require: Threshold δ
for i from 1 to λ **do**
  Initialize model h^i;
  Build set $\mathcal{D}_L^i = \{(x_1, p_1^i), ..., (x_n, p_n^i)\}$;
  Fit model h^i on $\mathcal{D}_L^i$;
end for
j=0;
while $\mathcal{D}_U$ is not empty **do**
  For each $x_k \in \mathcal{D}_U$, $\hat{p}_k = {}^{(h^1(x_k), ..., h^\lambda(x_k))}/_{(\sum_j h^j(x_k))}$;
  Build set $\mathcal{D}_j = \{x_k \in \mathcal{D}_U | \exists j \text{ s.t. } p_k^j \geq \delta\}$;
  for i from 1 to λ **do**
    Build set $\mathcal{D}_U^i = \{x_k | x_k \in \mathcal{D}_j\}$;
    One-hot encode all pseudo-labels on $\mathcal{D}_{U,j}^i$;
    $\mathcal{D}_T^i = \mathcal{D}_L^i \cup \mathcal{D}_U^i$;
    Fit model h^i on $\mathcal{D}_T^i$;
  end for
  $\mathcal{D}_{U,j} = \mathcal{D}_{U,j-1} \setminus \mathcal{D}_j$;
  j = j +1;
end while

Algorithm 4 Soft label strategy

Require: Labeled set $\mathcal{D}_L = \{(x_1, p_1), ..., (x_n, p_n)\}$, Unlabeled set $\mathcal{D}_U = \{(x_{n+1}), ..., (x_{n+m})\}$;
Require: Threshold δ.
for i from 1 to λ **do**
  Initialize model h^i;
  Build set $\mathcal{D}_L^i = \{(x_1, p_1^i), ..., (x_n, p_n^i)\}$;
  Fit model h^i on $\mathcal{D}_L^i$;
end for
j=0;
while $\mathcal{D}_U$ is not empty **do**
  For each $x_k \in \mathcal{D}_U$, $\hat{p}_k = {}^{(h^1(x_k), ..., h^\lambda(x_k))}/_{(\sum_j h^j(x_k))}$;
  Build set $\mathcal{D}_j = \{(x_k \in \mathcal{D}_U | \exists j \text{ s.t. } p_k^j \geq \delta\}$;
  for i from 1 to λ **do**
    Build set $\mathcal{D}_U^i = \{(x_k, \hat{p}_k^i | x_k \in \mathcal{D}_j)\}$;
    $\mathcal{D}_T^i = \mathcal{D}_L^i \cup \mathcal{D}_U^i$;
    Fit model h^i on $\mathcal{D}_T^i$;
  end for
  $\mathcal{D}_{U,j} = \mathcal{D}_{U,j-1} \setminus \mathcal{D}_j$;
  j = j +1;
end while

3.2 Results

Experiments were realized on the Wine, Ecoli, Digits, Poem and Iris datasets with 5 different seeds. We took into account

two metrics in order to compare the strategies: the classical accuracy and Expected Calibration Error (ECE). While the former is common, the latter is less used but it is interesting for our case since this metric measures how good is our calibration. ECE is computed by using the formula:

$$ECE = \sum_{m=1}^{n} \frac{|B_m|}{n} |acc(B_m) - conf(B_m)|, \qquad (11)$$

where we have M bins B_m, each with size $1/M$. Each B_m is the set of indices of samples whose prediction confidence falls into the interval $I_m = ((m-1)/M, m/M]$ and

$$acc(B_m) = \frac{\sum_{i \in B_m} \mathbb{1}_{\hat{y}_i = y_i}}{|B_m|}, \quad conf(B_m) = \frac{\sum_{i \in B_m} \hat{p}_i}{|B_m|}.$$

The ECE corresponds to an error, and should therefore be as low as possible [5].

The results are presented in Figure 3. Our credal strategy has comparable performances with respect to other methods, especially in the asymptotic regime. It takes more iterations to reach good performances, but it should be noted that we must save some data to perform the calibration: this means among other things that we start with less available labeled examples than other strategies. Thus, for smaller datasets, the calibration set $\mathcal{D}_C$ represents a higher percentage of all labeled data $\mathcal{D}_L$, which consequently decreases the percentage available for training.

With respect to ECE, we can notice a consistent low value in comparison to other strategies. This is expected as our strategy is the only one with calibration guarantees. Thus, our method is capable of having a comparable performance while obtaining a better calibration, hence giving similar performances but being trustworthier. Such a better calibration with respect to uncertainty quantification is also a key component if we want to adapt our strategy to other loss functions, as calibrated outputs are essential to compute reliable expected losses.

4 Conclusions and perspectives

In this paper, we investigated the use of credal labels in a self-learning procedure using Venn-Abers predictors and pairwise decomposition, in order to deal with the multi-class setting, while having fast, efficient and calibrated uncertainty estimates that could also model precise probabilities as a special case (other methods either consider possibility distributions or the binary case).

Our strategy is on par with other methods accuracy-wise, but exhibits a much better calibration in general. This seems of significant importance if we are concerned with training classifiers that deliver trustworhty uncertainty quantification. A possible way to improve our learning strategy would be, rather than using a fixed calibration data set, to slowly increase it with items from the initial $\mathcal{D}_L$ as we perform self-labeling. We could also imagine adding self-labels in the calibration data set, in the work of weakly supervised labels (such as considered in [3]).

References

[1] V. Cabannnes, A. Rudi, and F. Bach. Structured prediction with partial labelling through the infimum loss. In *International Conference on Machine Learning*, pages 1230–1239. PMLR, 2020.

[2] A. Campagner et al. Credal learning: Weakly supervised learning from credal sets. *FRONTIERS IN ARTIFICIAL INTELLIGENCE AND APPLICATIONS*, 372:327–334, 2023.

[3] M. Cauchois, S. Gupta, A. Ali, and J. C. Duchi. Predictive inference with weak supervision. *Journal of Machine Learning Research*, 25(118):1–45, 2024.

[4] L. M. De Campos, J. F. Huete, and S. Moral. Probability intervals: a tool for uncertain reasoning. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2 (02):167–196, 1994.

[5] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

[6] U. Johansson and P. Gabrielsson. Are traditional neural networks well-calibrated? In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019. doi: 10.1109/IJCNN.2019.8851962.

[7] J. Lienen and E. Hüllermeier. Credal self-supervised learning, 2021. URL https://arxiv.org/abs/2106.11853.

[8] V. Manokhin. Multi-class probabilistic classification using inductive and cross Venn–Abers predictors. In A. Gammerman, V. Vovk, Z. Luo, and H. Papadopoulos, editors, *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*, volume 60 of *Proceedings of Machine Learning Research*, pages 228–240. PMLR, 13–16 Jun 2017. URL https://proceedings.mlr.press/v60/manokhin17a.html.

[9] J. Rodemann, C. Jansen, G. Schollmeyer, and T. Augustin. In all likelihoods: How to reliably select pseudo-labeled data for self-training in semi-supervised learning. *arXiv preprint arXiv:2303.01117*, 2023.

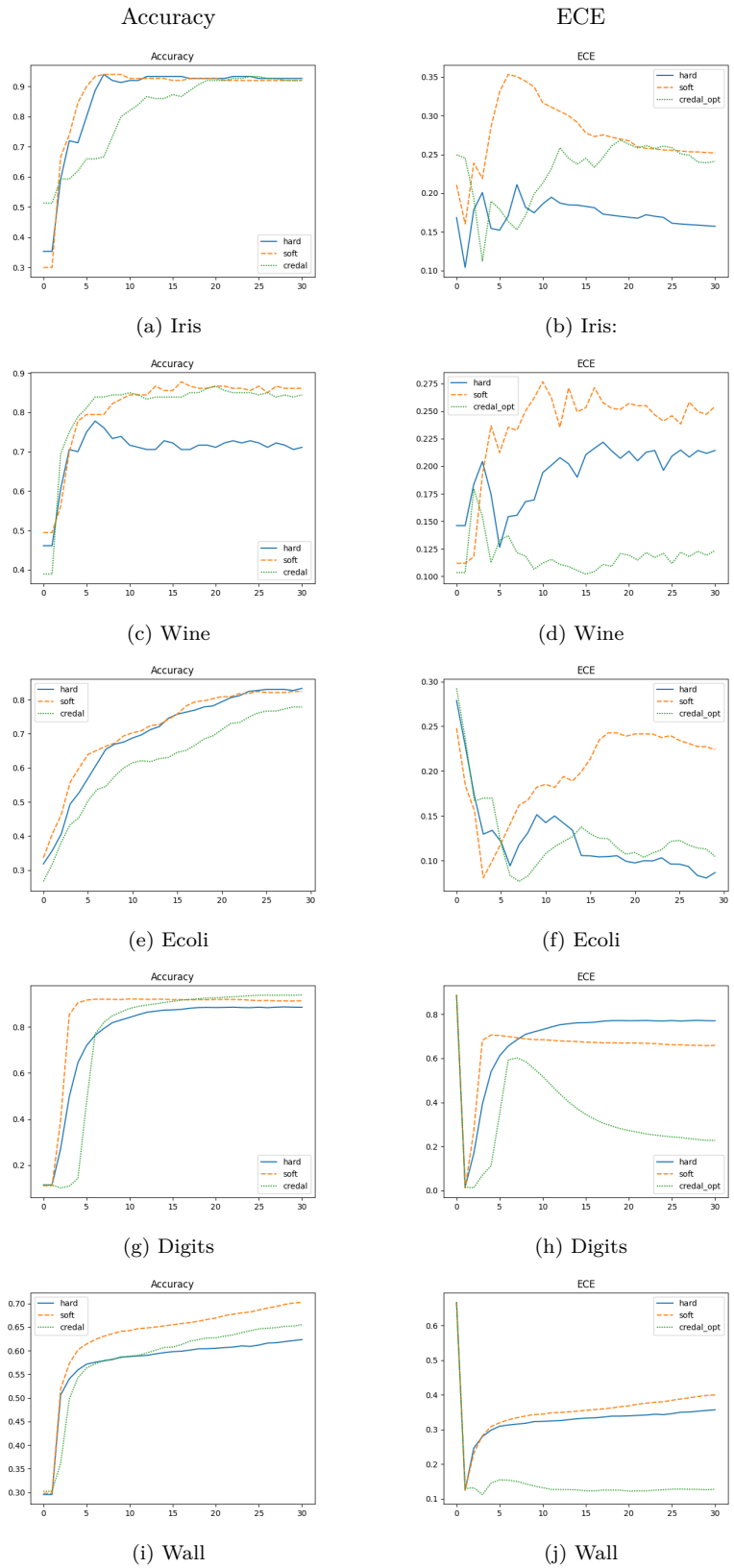
[10] C. Rodriguez, V. M. Bordini, S. Destercke, and B. Quost. Self learning using venn-abers predictors. In *Conformal and Probabilistic Prediction with Applications*, pages 234–250. PMLR, 2023.

[11] K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence, 2020. URL https://arxiv.org/abs/2001.07685.

[12] Y. Soullard, S. Destercke, and I. Thouvenin. Co-training with credal models. In *Artificial Neural Networks in Pattern Recognition: 7th IAPR TC3 Workshop, ANNPR 2016, Ulm, Germany, September 28–30, 2016, Proceedings 7*, pages 92–104. Springer, 2016.

[13] V. Vovk and I. Petej. Venn-abers predictors, 2012. URL https://arxiv.org/abs/1211.0025.

[14] P. Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman & Hall, 1991.



(a) Iris (b) Iris:

(c) Wine (d) Wine

(e) Ecoli (f) Ecoli

(g) Digits (h) Digits

(i) Wall (j) Wall

Figure 3: Accuracy and ECE for all datasets in order of dataset size, i.e, Iris being the smallest and Wall being the largest.