Hindawi Publishing Corporation
Computational Intelligence and Neuroscience
Volume 2015, Article ID 973696, 17 pages
http://dx.doi.org/10.1155/2015/973696

Research Article

Optimism in Active Learning

Timothé Collet[1,2] and Olivier Pietquin[3,4]

[1]*CentraleSupélec, MaLIS Research Group, 57070 Metz, France*
[2]*GeorgiaTech-CNRS UMI 2958, 57070 Metz, France*
[3]*Université de Lille-CRIStAL UMR 9189, SequeL Team, 59650 Villeneuve d'Ascq, France*
[4]*Institut Universitaire de France (IUF), 75005 Paris, France*

Correspondence should be addressed to Timothé Collet; timothe.collet@centralesupelec.fr

Received 15 April 2015; Accepted 12 August 2015

Academic Editor: Francesco Camastra

Active learning is the problem of interactively constructing the training set used in classification in order to reduce its size. It would ideally successively add the instance-label pair that decreases the classification error most. However, the effect of the addition of a pair is not known in advance. It can still be estimated with the pairs already in the training set. The online minimization of the classification error involves a tradeoff between exploration and exploitation. This is a common problem in machine learning for which multiarmed bandit, using the approach of Optimism int the Face of Uncertainty, has proven very efficient these last years. This paper introduces three algorithms for the active learning problem in classification using Optimism in the Face of Uncertainty. Experiments lead on built-in problems and real world datasets demonstrate that they compare positively to state-of-the-art methods.

1. Introduction

Traditional classification is a supervised learning framework in which the goal is to find the best mapping between an instance space and a label set. It is based only on the knowledge of a set of instances and their corresponding labels called the training set. To obtain it, an expert or oracle is required to manually label each of the examples, which is expensive. Indeed, this task is time consuming and may as well involve any other kind of resources. The aim of active learning [1] is to reduce the number of requests to the expert without losing performances, which is equivalent to maximizing the performance with a certain number of labeled instances. This can be done by dynamically constructing the training set. Each new instance presented to the expert is thus carefully chosen to generate the best gain in performance. The selection is guided by all the previous received labels. This is a sequential decision process [2].

However, the gain in performance due to a particular instance is not known in advance. This is for two reasons: first, the label given by the expert is not known before querying, and second, the true mapping is unknown. However, those values can be estimated also more and more precisely as the training set grows, because it is the goal of classification to get a good estimate of those values. Still, a low confidence must be put on the first estimations while later estimations may be more trusted. An instance may thus be presented to the expert because it is believed to increase the performances of the classifier, resulting in a short term gain. Or, because it will improve the estimations and help to select better instances in the future, resulting in a long term gain.

This is a very common problem in literature known as exploration versus exploitation dilemma. It has been successfully addressed under the multiarmed bandit problem, as introduced in [3] and surveyed in [4]. In this problem, a set of arms (choices) is considered, where each provides a stochastic reward when pulled (selected). The distribution of rewards for an arm is initially unknown. The goal is to define a strategy to successively pull arms, which maximizes the expected reward under a finite budget of pulls. Several methods have been introduced to solve this dilemma. One of them is the Upper Confidence Bound algorithm, introduced in [5]. It

uses the approach of Optimism in the Face of Uncertainty, which selects the arm for which the unknown expected reward is possibly the highest. One notable advantage of those algorithms is that they come with finite-sample analysis and theoretical bounds.

The idea is thus to use Optimism in the Face of Uncertainty for the Active Learning problem in classification. To use this approach, the problem is cast under the multiarmed bandit setting. However, this one deals with a finite number of arms, whereas in classification the instance space may be continuous. In order to adapt it to classification, the instance space is partitioned into several clusters. The goal is thus to find the best mapping between the clusters and the label set, under a finite budget of queries to the expert.

At first, we study the case of independent clusters, where the label given to each cluster only depends on the samples taken in it. We show two algorithms capable of the online allocation of samples among clusters. In this context, we need at least one (or even two) sample in each cluster in order to start favoring one for selection. Thus, the number of clusters must not be too high. This implies using a coarse partition which may limit the maximum performance. The choice of this partition is thus a key issue which has no obvious solution.

Allowing the prediction of each cluster to depend on the samples received in others enables us to use a more refined partition. This makes the choice of the partition less critical. We thus study the case of information sharing clusters. The adaptation of the first case to this one goes through the use of a set of coarse partitions combined by using a Committee of Experts approach. We introduce an algorithm that allocates samples in this context. Doing so, the number of clusters is not limited anymore, and increasing it allows us to apply our algorithms on a continuous instance space. Another algorithm is introduced as an extension of the first one using a kernel.

We start by an overview of the existing methods in active learning in Section 2. Then, in Sections 3–5, we describe the algorithms. We handle the cases of independent cluster and information sharing clusters. For each one of these problem we define a new loss function that has to be minimized. We also define the confidence interval used by our optimistic algorithms. In Section 6, we evaluate the performance of the algorithms in both built-in problems and real world datasets.

2. Related Work

Many algorithms already exist for active learning. A survey of those methods can be found in [6]. Among them, uncertainty sampling [7] uses a probabilistic classifier (it does not truly output a probability but a score on the label) and samples where the label to give is least certain. In binary classification with labels 0 or 1, this is where the score is closest to 0.5. Query by committee [8, 9] methods consider the version space or hypotheses space as the set of all consistent classifiers (nonnoisy classification) and try to reduce it as fast as possible by sampling the most discriminating instance. It finishes when only one classifier is left in the set. Extensions exist for the noisy case, either by requiring more samples before eliminating a hypothesis [10] or by associating a metric to the version space and trying to reduce it [11, 12]. Other algorithms exist that use a measure of confidence for the labels currently given, such as entropy [13] or variance [14]. Finally, the expected error reduction [15–18] algorithms come from the fact that the measure of performance is mostly the risk and that it makes more sense to minimize it directly rather than some other indirect criteria. Our work belongs to this last category. Using an optimistic approach enables us to minimize directly the true risk instead of the expected belief about it.

Other methods also use Optimism in the Face of Uncertainty for active learning. In [19], the method is more related to query by committee since it tries to select the best hypothesis from a set. It thus considers each hypothesis as an arm of a multiarmed bandit and plays them in an optimistic way. In [20], the authors study the problem of estimating uniformly well the mean values of several distributions under a finite budget. This is equivalent to the problem of active learning for regression with an independent discrete instance space. Although this algorithm may still be used on a classification problem, it is not designed for that purpose. Indeed, a good estimate of the mean values leads to a good prediction of the label. However, from the active learning point of view, it will spend effort to be precise on the estimation of the mean value even if this precision is of no use for the decision of the label. Efforts could have been spent to be more certain about the label to give. The importance of having an algorithm specifically adapted to classification is evaluated in Section 6.

3. Materials and Methods

The classical multiarmed bandit setting deals with a finite number of arms. This is not appropriate for the general classification problem in which the instance space may be continuous. In order to adapt this theory to active learning, we must first study the case of a discrete instance space, which may come from a discretized continuous space or originally discrete data. At first, we study the case of independent clusters, where no knowledge is shared between neighbors. After that, we will improve the selection strategy by letting neighbor clusters to share information. At the end, by defining clusters that contain only one instance from the pool each, with a good generalization behavior, we are able to apply this theory to continuous data. We may even define externally the relations between instances and use a kernel.

Let us define the following notations. We consider the instance space X and the label set Y. In binary classification, the label set is composed of two elements, in this work $Y = \{0, 1\}$. The oracle is represented by an unknown but fixed distribution $P(y \in Y \mid x \in X)$. The scenario considered in this work is pool-based sampling [7]. It assumes that there is a large pool of unlabeled instances available from which the selection strategy is able to pick. At each time step t, an active learning algorithm selects an instance $x_t \in X$ from the pool, receives a label $y_t \in Y$ drawn from the underlying distribution, and add the pair to the training set. This is repeated up to time n. The aim is to define a selection strategy that generates the best performance of the classifier at time n.

The performance is measured with the risk, which is the mean error that would achieve the classifier by predicting labels.

4. Independent Clusters

4.1. Partition of the Instance Space. In this section, we focus on the problem of defining a selection strategy with a discrete instance space. Either the space is already discrete or a continuous space is partitioned into several clusters. The following formulation assumes the latter case; otherwise, the same formulation applies for the discrete case if *clusters* are replaced with *instances*. The instance space is thus divided into K clusters. The problem is now to choose in which cluster to sample.

Let us define the partition

$$N = \{X_1, \ldots, X_K\} \tag{1}$$

with the following properties:

(i) $\forall i \in [\![1, K]\!] : X_i \neq \emptyset$, no cluster is empty,

(ii) $\cup_{i=1}^{K} X_i = X$, the clusters cover the whole instance space,

(iii) $\forall (i, j) \in [\![1, K]\!]^2 : i \neq j \Rightarrow X_i \cap X_j = \emptyset$, no clusters overlap.

It is important to note that the partition does not change during the progress of the algorithm.

Having discretized the instance space, we can now formalize the problem under a K-armed bandit setting. Each cluster $X_k \in N$ is an arm characterized by a Bernoulli distribution ν_k with mean value μ_k. Indeed, samples taken in a given cluster can only have a value of 0 or 1. At each round, or time step, $t \geq 1$, an allocation strategy selects an arm $k_t \in [\![1, K]\!]$, which corresponds to picking an instance randomly in the cluster X_{k_t} and receives a sample $y_{k,t} \sim \nu_k$, independently of the past samples. Let $(w_k)_{k \in [\![1,K]\!]}$ denote the weight of each cluster, with $\sum_{k=1}^{K} w_k = 1$. For example, in a semisupervised context using pool-based sampling, each weight is proportional to the number of unlabeled data points in each cluster, while, in membership query synthesis, the weights are the sizes or areas of clusters.

Let us define the following notations:

$$T_{k,t} = \sum_{s=1}^{t} \mathbb{1}_{k_s = k} \tag{2}$$

is the number of times arm k has been pulled up to time t and

$$\widehat{\mu}_{k,t} = \frac{1}{T_{k,t}} \sum_{s=1}^{T_{k,s}} y_{k,s} \tag{3}$$

is the empirical estimate of the mean μ_k at time t.

Under this partition, the mapping of the instance space to the label set is limited to the mapping of clusters to the label set. We thus define the classifier that creates this mapping according to the samples received up to time t. In this section, the clusters are assumed to be independent. This means that the label given to a cluster can only depend on samples in this cluster. We use the naive Bayes classifier that gives the label

$$f(x \in X_k) = l_{k,t} = [\widehat{\mu}_{k,t}] \tag{4}$$

to cluster k, where $[\cdot]$ is the round operator.

4.2. Full Knowledge Criteria. The goal is to build an optimist algorithm for the active learning problem. A common methodology in the Optimism in the Face of Uncertainty paradigm is to characterize first the optimal solution. We thus place ourselves in the Full Knowledge setting. In this setting, we let the allocation strategy depend on the true value of μ_k for each cluster, and this defines the optimal allocation of the budget n. An optimist algorithm will then estimate those values and allocate samples as close as possible to the optimal allocation. Note that the true values of μ_k cannot be used by the classifier directly but only by the allocation strategy.

In the following sections, we show two full knowledge criteria: data-dependent and data-independent. In the data-independent case, the optimal allocation does not depend on the samples received so far. It can be related to one-shot active learning, as defined in [18], in which the allocation of the budget is decided before sampling any instances. In the data-dependent case, the label given by the classifier at time t is also considered. This is related to fully sequential active learning, as defined in [18], where the allocation of the budget is updated after each sample. Note that in both cases, the optimist algorithms built upon those criteria are fully sequential.

4.2.1. Data-Independent Criterion. In this section, we characterize the optimal allocation of the budget n depending only on the values of μ_k for each cluster. We want an allocation of the budget that minimizes the true risk of the classifier at time n. Here, the risk is based on the binary loss:

$$L_{0/1}(y, f(x)) = \begin{cases} 1, & \text{if } f(x) \neq y, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Note that this loss is usually hard to use because of its nonconvex nature.

Using the partition N, the true risk of the classifier is the sum of the true risks in each cluster

$$R(f) = \sum_{k=1}^{K} w_k R_k(f) = \sum_{k=1}^{K} w_k R_k(l_{k,n}) \tag{6}$$

with

$$R_k(l_{k,n}) = \begin{cases} 1 - \mu_k, & \text{if } l_{k,n} = 1, \\ \mu_k, & \text{if } l_{k,n} = 0. \end{cases} \tag{7}$$

The risk is the mean number of misclassified instances resulting from a particular prediction of labels.

The optimal label the algorithm should assign to arm k is $[\mu_k]$. This incurs a regret in the true risk $R_k([\widehat{\mu}_{k,n}]) - R_k([\mu_k])$. In order to define an allocation of the samples according

to the $[\mu_k]$ values regardless of their estimates, the regret is expected over all the samples. This gives us the following definition of the loss for classification per cluster, as the expected regret of the true risk in each cluster,

$$L_{k,n}(\mu_k, T_{k,n}) = \mathbb{E}\left[R_k\left([\widehat{\mu}_{k,n}]\right) - R_k\left([\mu_k]\right)\right], \tag{8}$$

where the expectation is taken over the samples:

$$\begin{aligned} L_{k,n}(\mu_k, T_{k,n}) &= \mathbb{E}\left[(1-\mu_k)\,\mathbb{1}_{[\widehat{\mu}_{k,n}]=1} + \mu_k \mathbb{1}_{[\widehat{\mu}_{k,n}]=0}\right. \\ &\quad \left. - (1-\mu_k)\,\mathbb{1}_{[\mu_k]=1} - \mu_k \mathbb{1}_{[\mu_k]=0}\right] = 2\left|\mu_k - 0.5\right| \\ &\quad \cdot \mathbb{P}\left([\widehat{\mu}_{k,n}] \neq [\mu_k]\right). \end{aligned} \tag{9}$$

The value to be minimized by our allocation of the budget is then the global loss. It is the sum of losses in each cluster:

$$L_n\left((\mu_k)_{k\in[\![1,K]\!]}, (T_{k,n})_{k\in[\![1,K]\!]}\right) = \sum_{k=1}^{K} w_k L_{k,n}(\mu_k, T_{k,n}). \tag{10}$$

The objective is now to define an allocation of the budget that minimizes this loss. However, in order to inverse the loss to retrieve the allocation, as well as to derive the online allocation strategy, the losses in each cluster have to be strictly decreasing with $T_{k,t}$ and convex. This is not the case with these losses. In order to get a more convenient shape, we bound those losses by pseudolosses. The algorithms we build aim to minimize this pseudoloss instead of the loss defined previously. The idea is thus to bound the probability $\mathbb{P}([\widehat{\mu}_{k,t}] \neq [\mu_k])$. We use the fact that the estimated mean in one subset follows a binomial distribution (labels are either 0 or 1). The bounds obtained this way are very tight and equal at a infinitely countable number of points.

Let $\mathcal{J}_{1-p}(n - \lfloor k \rfloor, \lfloor k \rfloor + 1)$ be the cumulative distribution function of a binomial distribution of parameters n, p. Then,

$$\begin{aligned} &\mathbb{P}\left([\widehat{\mu}_{k,n}] \neq [\mu_k]\right) \\ &= \mathbb{1}_{[\mu_k]=0}\mathbb{P}\left(\widehat{\mu}_{k,n} \geq 0.5\right) + \mathbb{1}_{[\mu_k]=1}\mathbb{P}\left(\widehat{\mu}_{k,n} < 0.5\right) \\ &= \mathbb{1}_{[\mu_k]=0}\left(1 - \mathcal{J}_{1-\mu_k}\left(T_{k,n} - \left\lfloor \frac{T_{k,n}}{2} \right\rfloor, \left\lfloor \frac{T_{k,n}}{2} \right\rfloor + 1\right)\right) \\ &\quad + \mathbb{1}_{[\mu_k]=1}\mathcal{J}_{1-\mu_k}\left(T_{k,n} - \left\lfloor \frac{T_{k,n}}{2} \right\rfloor, \left\lfloor \frac{T_{k,n}}{2} \right\rfloor + 1\right). \end{aligned} \tag{11}$$

Note that the probability given above is a step function of $T_{k,n}/2$ and thus is not a strictly decreasing function of $T_{k,n}$. That is not convenient as we require this condition in the later. That is why we bound this probability by bounding the truncated value $\lfloor T_{k,n}/2 \rfloor$. Then,

$$\begin{aligned} &\mathbb{P}\left([\widehat{\mu}_{k,n}] \neq [\mu_k]\right) \\ &\leq \mathbb{1}_{[\mu_k]=0}\left(1 - \mathcal{J}_{1-\mu_k}\left(\frac{T_{k,n}}{2} + 1, \frac{T_{k,n}}{2}\right)\right) \\ &\quad + \mathbb{1}_{[\mu_k]=1}\mathcal{J}_{1-\mu_k}\left(\frac{T_{k,n}}{2}, \frac{T_{k,n}}{2} + 1\right). \end{aligned} \tag{12}$$

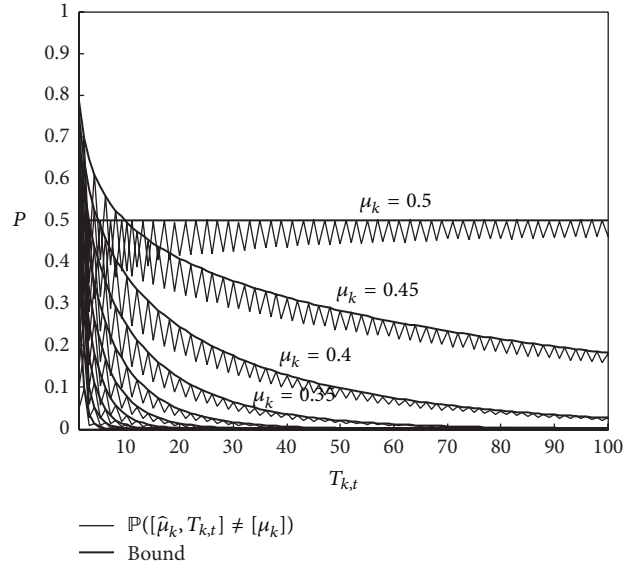


Figure 1: $\mathbb{P}([\widehat{\mu}_{k,t}] \neq [\mu_k])$ and its bound defined in (12).

Figure 1 displays this probability and the corresponding bound function of $T_{k,t}$ for different values of μ_k. We can see that the bound is extremely tight, and its only role is to make it strictly decreasing with $T_{k,t}$ and convex. It still retains as much as possible the shape of the probability.

We therefore define the following pseudoloss:

$$\widetilde{L}_n\left((\mu_k)_{k\in[\![1,K]\!]}, (T_{k,n})_{k\in[\![1,K]\!]}\right) = \sum_{k=1}^{K} \widetilde{L}_{k,n}(\mu_k, T_{k,n}) \tag{13}$$

with

$$\begin{aligned} \widetilde{L}_{k,n}(\mu_k, T_{k,n}) &= 2w_k\left|\mu_k - 0.5\right| \\ &\quad \cdot \left[\mathbb{1}_{[\mu_k]=0}\left(1 - \mathcal{J}_{1-\mu_k}\left(\frac{T_{k,n}}{2} + 1, \frac{T_{k,n}}{2}\right)\right)\right. \\ &\quad \left. + \mathbb{1}_{[\mu_k]=1}\mathcal{J}_{1-\mu_k}\left(\frac{T_{k,n}}{2}, \frac{T_{k,n}}{2} + 1\right)\right] \end{aligned} \tag{14}$$

being the pseudoloss in each cluster.

Due to the convex nature of $\widetilde{L}_{k,n}$, $\partial\widetilde{L}_{k,n}/\partial T_{k,n}$ is a strictly increasing function of $T_{k,n}$. Thus, it admits an inverse $(\partial\widetilde{L}_{k,n}/\partial T_{k,n})^{-1}$.

Let $T^*_{k,n}$ be the optimal number of samples to take in each subset in order to minimize $\widetilde{L}_n$ under the constraint that $\sum_{k=1}^{K} T^*_{k,n} = n$:

$$T^*_{k,n} = \left(\frac{\partial\widetilde{L}_{k,n}}{\partial T_{k,n}}\right)^{-1}(\mu_k c^*) \tag{15}$$

with c^* such that $\sum_{k=1}^{K}(\partial\widetilde{L}_{k,n}/\partial T_{k,n})^{-1}(\mu_k, c^*) = n$.

This defines the theoretical optimal allocation of the budget. Since we do not know the closed form for $(\partial\widetilde{L}_{k,n}/\partial T_{k,n})^{-1}$

and since an optimist algorithm needs an online allocation criterion, we now show the online allocation criterion $C_{k,t}$,

$$C^i_{k,t}\left(\mu_k, T_{k,t}\right) = -\frac{\partial \widetilde{L}_{k,n}}{\partial T_{k,n}}\left(\mu_k, T_{k,t}\right), \tag{16}$$

is such that an algorithm sampling at each time t the cluster X_{k_t} with

$$k_t \in \arg\max_{1\le k\le K} \frac{T^*_{k,t}}{T_{k,t}} = \arg\max_{1\le k\le K} C^i_{k,t}\left(\mu_k, T_{k,t}\right) \tag{17}$$

would result in the optimal allocation of the budget n.

We have seen here an optimal allocation of the budget n that the optimist algorithm which will be defined in Section 4.3 could try to reach without the knowledge of the μ_k values. The criterion we derived only depends on the values of the parameters in each cluster and not the current labels given by the classifier. Considering them would lead to a better allocation since the allocation in a cluster could stop when the correct label is given.

4.2.2. Data-Dependent Criterion. In this section, we show a criterion that leads to the optimal allocation of the budget n depending not only on the values of μ_k in each cluster, but also on the current labels given by the classifier.

We define a new global loss that is the current regret of the true risk:

$$\begin{aligned} &L^d_n\left(\left(\mu_k\right)_{k\in[\![1,K]\!]}, \left(\widehat{\mu}_{k,n}\right)_{k\in[\![1,K]\!]}, \left(T_{k,n}\right)_{k\in[\![1,K]\!]}\right) \\ &\quad = \sum_{k=1}^{K} w_k L^d_{k,n}\left(\mu_k, \widehat{\mu}_{k,n}, T_{k,n}\right), \end{aligned} \tag{18}$$

with

$$\begin{aligned} L^d_{k,n}\left(\mu_k, \widehat{\mu}_{k,n}, T_{k,n}\right) &= R_k\left(\left[\widehat{\mu}_{k,n}\right]\right) - R_k\left(\left[\mu_k\right]\right) \\ &= 2\left|\mu_k - 0.5\right| \mathbb{1}_{\left[\widehat{\mu}_{k,n}\right]\neq\left[\mu_k\right]}. \end{aligned} \tag{19}$$

The measure of performance is still the expected true risk but the value to be minimized is preferred to be run-dependent.

In order to minimize it, the selection strategy samples the cluster for which the expected decrease of the loss would be maximum. This criterion is thus the finite difference of the loss $\mathbb{E}[\Delta L^d_{k,t}]$ with

$$\begin{aligned} \Delta L^d_{k,t} = L^d_{k,t}&\left(\mu_k, \frac{T_{k,t}\widehat{\mu}_{k,t} + s}{T_{k,t}+1}, T_{k,t}+1\right) \\ &- L^d_{k,t}\left(\mu_k, \widehat{\mu}_{k,t}, T_{k,t}\right), \end{aligned} \tag{20}$$

where s is the label resulting from the sample and the expectation is taken on s.

However, this is a good strategy only if this criterion is strictly increasing with $T_{k,n}$. We thus study the monotonicity of this criterion. We consider sampling T^+ more instances in cluster k with resulting average label $\widehat{\mu}^+$. The new label given by the classifier will be $l^+_{k,t} = [(T_{k,t}\widehat{\mu}_{k,t} + T^+\widehat{\mu}^+)/(T_{k,t}+T^+)]$.

After T^+ samples, the expected decrease of the loss is

$$\begin{aligned} &\mathbb{E}\left[\Delta_{T^+} L^d_{k,t}\right] \\ &\quad = 2\left|\mu_k - 0.5\right|\left(\mathbb{P}\left(l^+_{k,t} \neq \left[\mu_k\right]\right) - \mathbb{1}_{\left[\widehat{\mu}_{k,n}\right]\neq\left[\mu_k\right]}\right). \end{aligned} \tag{21}$$

Injecting the value of $l^+_{k,t}$,

$$\begin{aligned} \mathbb{P}\left(l^+_{k,t}=1\right) &= \mathbb{P}\left(\frac{T_{k,t}\widehat{\mu}_{k,t} + T^+\widehat{\mu}^+}{T_{k,t}+T^+} \ge 0.5\right) \\ &= \mathbb{P}\left(T^+\widehat{\mu}^+ \ge T_{k,t}\left(0.5 - \widehat{\mu}_{k,t}\right) + \frac{T^+}{2}\right). \end{aligned} \tag{22}$$

To shorten notations we use $D_{k,t} = 2T_{k,t}(0.5 - \widehat{\mu}_{k,t})$.

We know that $\widehat{\mu}^+$ is drawn from a binomial distribution of parameter μ_k and T^+, thus

$$\begin{aligned} \mathbb{P}\left(l^+_{k,t}=1\right) &= 1 - \mathscr{J}_{1-\mu_k}\left(T^+ - \left\lfloor\frac{D_{k,t}+T^+}{2}\right\rfloor, \left\lfloor\frac{D_{k,t}+T^+}{2}\right\rfloor + 1\right), \\ \mathbb{P}\left(l^+_{k,t}=0\right) &= \mathscr{J}_{1-\mu_k}\left(T^+ - \left\lfloor\frac{D_{k,t}+T^+}{2}\right\rfloor, \left\lfloor\frac{D_{k,t}+T^+}{2}\right\rfloor + 1\right). \end{aligned} \tag{23}$$

The criterion is not strictly increasing. In order to consider this constraint, we define another criterion which is a tight bound of the previous one. We first bound the following probabilities:

$$\begin{aligned} \mathbb{P}\left(l^+_{k,t}=1\right) &\le 1 - \mathscr{J}_{1-\mu_k}\left(\frac{T^+ - D_{k,t}}{2} + 1, \frac{T^+ + D_{k,t}}{2}\right) \\ &= P_1\left(\mu_k, T^+, T_{k,t}, \widehat{\mu}_{k,t}\right). \end{aligned} \tag{24}$$

Equivalently,

$$\begin{aligned} &\mathbb{P}\left(l^+_{k,t}=0\right)\mathbb{P}\left(l^+_{k,t}=0\right) \\ &\quad \le \mathscr{J}_{1-\mu_k}\left(\frac{T^+ - D_{k,t}}{2}, \frac{T^+ + D_{k,t}}{2} + 1\right) \\ &\quad = P_0\left(\mu_k, T^+, T_{k,t}, \widehat{\mu}_{k,t}\right). \end{aligned} \tag{25}$$

The criterion resulting from this bounds is strictly increasing but is not defined for all T^+. Indeed, in order to change the value of the label, the estimated mean has to move to the other side of 0.5. This often requires more than one sample (e.g., if we already sampled 10 instances and 8 were labeled 1, we need at least 6 new samples to have a chance to change the label given by the classifier). In order to get a bound defined for $T^+ = 1$ and strictly increasing with T^+, we make a linear interpolation between the value in $T^+ = |D_{k,t}|$ and the value in $T^+ = 0$ which is 0.

We thus define the actual criterion:

$$\widetilde{\Delta L}^{d}_{k,t} = \frac{2w_k \left|\mu_k - 0.5\right|}{\left|D_{k,t}\right|} \left(\mathbb{1}_{[\mu_k]=0} P_0\left(\mu_k, \left|D_{k,t}\right|, T_{k,t}, \widehat{\mu}_{k,t}\right) + \mathbb{1}_{[\mu_k]=1} P_0\left(\mu_k, \left|D_{k,t}\right|, T_{k,t}, \widehat{\mu}_{k,t}\right) - \mathbb{1}_{[\widehat{\mu}_{k,t}]\neq[\mu_k]} \right). \tag{26}$$

The online allocation criterion is

$$C^{d}_{k,t}\left(\mu_k, T_{k,t}, \widehat{\mu}_{k,t}\right) = -\widetilde{\Delta L}^{d}_{k,t}, \tag{27}$$

and it is such that an algorithm sampling at each time t the cluster X_{k_t} with

$$k_t \in \arg\max_{1\leq k\leq K} C^{d}_{k,t}\left(\mu_k, T_{k,t}, \widehat{\mu}_{k,t}\right) \tag{28}$$

would result in the optimal allocation of the budget n.

The criterion defined in this section leads to an optimal allocation of the budget n that the optimist algorithm which will be defined in the next section could try to reach without the knowledge of the μ_k values. It depends on the value of the parameters in each cluster as well as the current estimate of this parameter by the classifier.

4.3. Included Optimism. In this section we introduce two optimistic algorithms: OALC-DI (Optimistic Active Learning for Classification: Data Independent) which use the data-independent criterion and OALC-DD (Optimistic Active Learning for Classification: Data Dependent) which use the data-dependent criterion for optimal budget allocation defined in the previous sections. Both can be described by the same core algorithm. Neither criteria can be used as they are currently defined, for the active learning problem. Indeed, the value of μ_k in each cluster is not known in advance; otherwise, the correct label would be known as well. Also, it cannot directly replace those values by their estimation which could lead to clusters being turned down. This is a case of the exploration/exploitation tradeoff where the uncertainty about the true value of μ_k in each cluster has to be considered. Therefore, we design an optimistic algorithm that estimates those values and samples as close as possible to the optimal allocation.

Following the Optimism in the Face of Uncertainty approach, it builds a confidence interval on the criterion to be maximized and draw the arm for which the upper bound of this interval is highest. This is equivalent to saying it draws the arm for which the criterion is possibly the highest. As we know the shape of the distribution of the $\widehat{\mu}_{k,t}$ values, the confidence interval is a Bayesian Credible Interval [21] which leads to tight bounds. The Bayesian Credible Interval is relative to a probability δ which allows for controling the amount of exploration of the algorithm. The core algorithm is presented in Algorithm 1. It takes one parameter δ and can be derived in two algorithms depending on the criterion used.

Let us show how to build the Bayesian Credible Interval. As each sample is drawn from a Bernoulli distribution, the estimated means follow a binomial distribution. Beta distributions provide a family of conjugate prior probability distributions for binomial distributions. The uniform distribution Beta(1, 1) is taken as the prior probability distribution, because we have no information about the true distribution. Using the Bayesian inference,

Input: δ
Intitialize: Sample each cluster once
for $t = K+1, \ldots, n$ **do**
  Compute $e_{k,t}$ from (32) with $C_{k,t}$ from (16) or (27)
  Sample the cluster X_{k_t} with $k_t = \arg\max_k e_{k,t}$
end
Output: $[\widehat{\mu}_{k,n}]$ for all arms $1 \leq k \leq K$

Algorithm 1: Core algorithm.

$$\mathbb{P}\left(\mu_k = x \mid \widehat{\mu}_{k,t}, T_{k,t}\right) = \frac{x^{T_{k,t}\widehat{\mu}_{k,t}} (1-x)^{T_{k,t}(1-\widehat{\mu}_{k,t})}}{\mathrm{Beta}\left(T_{k,t}\widehat{\mu}_{k,t}+1, T_{k,t}\left(1-\widehat{\mu}_{k,t}\right)+1\right)}. \tag{29}$$

In the following $C_{k,t}$ means either $C^{i}_{k,t}$ from (16) or $C^{d}_{k,t}$ from (27). Obviously,

$$\mathbb{P}\left(C_{k,t} > e_k \mid \widehat{\mu}_{k,t}, T_{k,t}\right) = \mathbb{P}\left(\mu_k, C_{k,t} > e_k \mid \widehat{\mu}_{k,t}, T_{k,t}\right). \tag{30}$$

Let $I_k = \{\mu_k \mid f(T_{k,t}, \mu_k) > e_k\}$, then

$$\mathbb{P}\left(C_{k,t} > e_k \mid \widehat{\mu}_{k,t}, T_{k,t}\right) = \frac{\int_{x\in I_k} x^{T_{k,t}\widehat{\mu}_{k,t}} (1-x)^{T_{k,t}(1-\widehat{\mu}_{k,t})}\, dx}{\mathrm{Beta}\left(T_{k,t}\widehat{\mu}_{k,t}+1, T_{k,t}\left(1-\widehat{\mu}_{k,t}\right)+1\right)}. \tag{31}$$

The upper bound of the Bayesian Credible Interval is then

$$e_k \text{ s.t. } \mathbb{P}\left(C_{k,t} > e_k \mid \widehat{\mu}_{k,t}, T_{k,t}\right) = \delta. \tag{32}$$

In this section, we have shown two optimistic algorithms that share the same core. The difference lies in the full knowledge criterion used. One depends only on the value of the parameters of the distributions. The other one depends on both the value of the parameters and the current estimates of this parameter by the classifier. Both the resulting algorithms depend only on the estimates of the parameters.

The problem solved by those algorithm is the one that finds the best label to give to several separated clusters. This separation comes from the partition of a continuous instance space. A good hypothesis would be that the μ_k values do not vary fast and that neighbor clusters have close values of μ_k. In order to speed up learning and to increase generalization, we could estimate μ_k considering neighbor clusters. This is the subject of next section.

5. Information Sharing Clusters

5.1. A Set of Partitions. The previous section introduces an active learning algorithm which is based on a partition of the instance space. Supposing this partition is given, it defines the

best allocation of samples among its clusters that lead to the lowest true risk of the classifier also based on this partition. The best performance of the classifier still highly depends on the choice of the partition, which has no obvious solution. One way to improve the classifier's performance is to increase the number of clusters in the partition. But this slows learning as each cluster parameter has to be estimated independently. To counter that, we allow the classifier to generalize by letting neighbor clusters share information. In order to use the same approach as before, we consider the case of a committee of partitions. Each partition estimates the parameter of their clusters independently. Then, the local prediction of the label is determined by averaging the estimations of each partition.

Let $\mathbf{N}$ be a set of m partitions of the instance space:

$$\mathbf{N} = \{N_1, \ldots, N_m\}, \tag{33}$$

where $\forall j \in \{1, \ldots, m\}$:

$$N_j = \left\{X_{1,j}, \ldots, X_{K_j,j}\right\}, \tag{34}$$

with the following properties:

(i) $\forall i \in [\![1, K_j]\!] : X_{i,j} \neq \emptyset$; no subset is empty,

(ii) $\cup_{i=1}^{K_j} X_{i,j} = X$; the subsets cover the whole instance space,

(iii) $\forall (i,k) \in [\![1, K_j]\!]^2 : i \neq k \Rightarrow X_{i,j} \cap X_{k,j} = \emptyset$; no subsets overlap.

Each partition may have a different number of subsets K_j.

These partitions may come from random forests [22] or tile coding which is a function approximation method commonly used in the field of reinforcement learning [23]. The partitions must not change during the progress of the algorithm.

We write $\widehat{\mu}_{i,j} = (1/W_{i,j}) \sum_{t=1}^{n} \mathbb{1}_{x_t \in X_{i,j}} y_t$, the average label in each cluster of each partition, and $W_{i,j} = \sum_{t=1}^{n} \mathbb{1}_{x_t \in X_{i,j}}$, the number of samples in subset j.

Let us now define the thinnest partition $\mathcal{N}$, which is the partition resulting from overlapping all the partitions from $\mathbf{N}$:

$$\mathcal{N} = \{\mathcal{X}_1, \ldots, \mathcal{X}_{\mathcal{K}}\} \tag{35}$$

such that $\forall c \in [\![1, \mathcal{K}]\!]$, $\forall (x_a, x_b) \in \mathcal{X}_c^2 : \forall j \in [\![1, m]\!]$, $\exists i \in [\![1, K_j]\!]$, $x_a \in X_{i,j} \Leftrightarrow x_b \in X_{i,j}$.

This means that two elements coming from the same subset of the thinnest partition necessarily come from the same subset in any partition of the set.

Each cluster c of this thinnest partition is associated with a Bernoulli distribution of parameter μ_c.

We write $\widehat{\mu}_c = (1/W_c) \sum_{t=1}^{n} \mathbb{1}_{x_t \in \mathcal{X}_c} y_t$, the average label in cluster c, and $W_c = \sum_{t=1}^{n} \mathbb{1}_{x_t \in \mathcal{X}_c}$, the number of samples in cluster c of the thinnest partition. We write $W_{c,i,j} = W_c / \sum_{c=1}^{\mathcal{K}} \mathbb{1}_{\mathcal{X}_c \subset X_{i,j}} W_c$, the relative importance of cluster c of the thinnest partition in cluster i of partition j. Note that $\widehat{\mu}_{i,j} = \sum_{c=1}^{\mathcal{K}} \mathbb{1}_{\mathcal{X}_c \subset X_{i,j}} W_{c,i,j} \widehat{\mu}_c$.

In order to understand what those values exactly are, an illustration is given on Figure 2. It shows an example

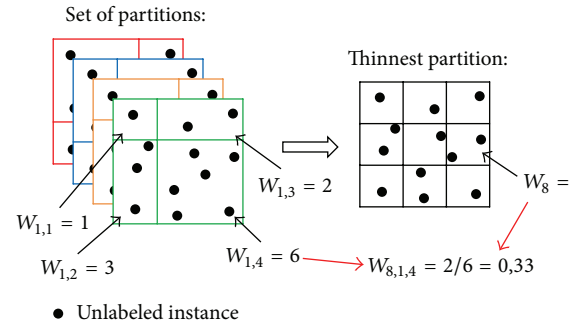


Figure 2: Representation of a set of partitions for a 2D instance space.

of the partition of a two-dimensional instance space. The set is composed of four partitions of four clusters each. The resulting thinnest partition is, in this case, composed of nine clusters. The dots represent the unlabeled instances with which the weights of each clusters of each partition are computed. The influence of a cluster of the thinnest partition in the estimation of the mean value in one cluster of a specific partition of the set is also computed. What is not shown is how the influence of this last estimation on the final prediction is computed. It is 1/4 in this case because there are 4 partitions in the set.

The prediction of the label is cluster c that results from the averaging of estimations of each partition:

$$l_c = \left[\frac{1}{m} \sum_{j=1}^{m} \sum_{i=1}^{K_j} \mathbb{1}_{\mathcal{X}_c \subset X_{i,j}} \widehat{\mu}_{i,j}\right]. \tag{36}$$

This can also be written as

$$(l_1, \ldots, l_{\mathcal{K}})^T = \left[P \times (\widehat{\mu}_1, \ldots, \widehat{\mu}_{\mathcal{K}})^T\right], \tag{37}$$

where the elements of P are $\forall (c_1, c_2) \in [\![1, \mathcal{K}]\!]^2$

$$P_{c_1,c_2} = \left[\frac{1}{m} \sum_{j=1}^{m} \sum_{i=1}^{K_j} \mathbb{1}_{\mathcal{X}_{c_1} \subset X_{i,j}} \mathbb{1}_{\mathcal{X}_{c_2} \subset X_{i,j}} W_{c_2,i,j}\right]. \tag{38}$$

Note that the size of the matrix P depends on the number of subsets in the thinnest partition. It may be very large if the initial partitions are not constrained. This is not a problem since a subset of the thinnest partition containing no instance from the training set causes its corresponding column of P to be null. It can therefore be removed. The size of P is thus limited by the number of instances in the training set.

In the active learning setting with a pool-based sampling scheme, the labels of the instances in the training set are not known in advance but are acquired sequentially. Although the pool of unlabeled instances is known initially, this allows us to compute the matrix P at the beginning of our algorithm and keep it until the end. This is different from the case where we consider only data already sampled, as done in Random Forests, and the matrix P has to be recomputed at each step.

Each cluster $\mathcal{X}_c \in \mathcal{N}$ is an arm of a multiarmed bandit characterized by a Bernoulli distribution ν_c with mean value

μ_c. At each round, or time step, $t \geq 1$, an allocation strategy selects an arm $c_t \in [\![1, \mathscr{K}]\!]$, which corresponds to picking an instance randomly in the cluster $\mathscr{X}_{c_t}$ and receives a sample $y_t \sim \nu_c$, independently of the past samples. $(W_c)_{c \in [\![1,\mathscr{K}]\!]}$ denote the weight of each cluster.

Then, if $T_{c,t} = \sum_{s=1}^{t} \mathbb{1}_{x_s \in \mathscr{X}_c}$ is the number of samples taken in subset c up to time t and $\widehat{\mu}_{c,t} = (1/T_{c,t}) \sum_{s=1}^{t} \mathbb{1}_{x_s \in \mathscr{X}_c} y_s$ if $T_{c,t} \neq 0$ and 0.5, otherwise, is the mean of labels taken up to time t.

Let us define the classifier that gives to subset c the label

$$f\left(x \in \mathscr{X}_c\right) = l_{c,t} = \left[\frac{\sum_{c'=1}^{\mathscr{K}} P_{c,c'} \widehat{\mu}_{c',t} - 0.5}{\sum_{c'=1}^{\mathscr{K}} P_{c,c'} \mathbb{1}_{T_{c'}>0}} + 0.5 \right]. \quad (39)$$

Note that it gives the same label as (37) where $l_{c,t} = [\sum_{c'=1}^{\mathscr{K}} P_{c,c'} \widehat{\mu}_{c',t}]$, and the only difference is that the value inside the $[\cdot]$ is reweighted such that it is 1 when all $\widehat{\mu}_{c,t}$ are equal to 1.

5.2. Full Knowledge Criterion. In this section, we define the allocation of the budget in the full knowledge setting for the case of information sharing clusters. We also introduce a criterion that leads to this allocation. Again, those parameters are only used to define the allocation of the budget and not for the prediction of labels.

In this problem, the clusters are not independent. This means sampling an instance in a cluster affects the prediction of other clusters. The criterion has shown here the results of the myopic minimization of the true risk. With the weights of each clusters being estimated from the number of instances in the pool (labeled and unlabeled), the true risk is computed as the risk on the pool. To decide the next cluster to sample, we simulate sampling in each cluster and evaluate its expected impact on the risk. The selected cluster is thus the one which lowers the risk most.

Here, the risk is based on the binary loss:

$$L_{0/1}\left(y, f\left(x\right)\right) = \begin{cases} 1, & \text{if } f\left(x\right) \neq y, \\ 0, & \text{otherwise.} \end{cases} \quad (40)$$

Note that this loss is usually hard to use because of its nonconvex nature.

First, suppose that the label $l_{c,t}$ is given to each subset by the classifier at time t. The true risk encountered in each subset is

$$R_c\left(l_{c,t}\right) = \begin{cases} W_c\left(1 - \mu_c\right), & \text{if } l_{c,t} = 1, \\ W_c \mu_c, & \text{if } l_{c,t} = 0. \end{cases} \quad (41)$$

Note that the best risk is attained when $l_{c,t} = [\mu_c]$.

Then, the true risk in each subset can also be written as follows:

$$R_c\left(l_{c,t}\right) = W_c \left|\mu_c - 0.5\right| \mathbb{1}_{l_{c,t} \neq [\mu_c]} + R_c\left(\left[\mu_c\right]\right). \quad (42)$$

Let $\mathbf{T}_t = (T_{1,t}, \ldots, T_{\mathscr{K},t})^T$ be the number of samples taken and $\widehat{\boldsymbol{\mu}}_t = (\widehat{\mu}_{1,t}, \ldots, \widehat{\mu}_{\mathscr{K},t})^T$ the current mean of labels, in each subset at time t. Let P_c be the cth row of P.

Let us remember that the labels given by the classifier come from (39)

$$R_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \mu_c\right) = W_c \left|\mu_c - 0.5\right| \mathbb{1}_{[P_c \times \widehat{\boldsymbol{\mu}}_t] \neq [\mu_c]} + R_c\left(\left[\mu_c\right]\right). \quad (43)$$

Let us note

$$\mathbf{T}_t^c = \left(T_{1,t}, \ldots, T_{c-1,t}, T_{c,t} + 1, T_{c+1,t}, \ldots, T_{\mathscr{K},t}\right)^T,$$

$$\widehat{\boldsymbol{\mu}}_t^{c+} = \left(\widehat{\mu}_{1,t}, \ldots, \widehat{\mu}_{c-1,t}, \frac{T_{c,t}\widehat{\mu}_{c,t} + 1}{T_{c,t} + 1}, \widehat{\mu}_{c+1,t}, \ldots, \widehat{\mu}_{\mathscr{K},t}\right)^T, \quad (44)$$

$$\widehat{\boldsymbol{\mu}}_t^{c-} = \left(\widehat{\mu}_{1,t}, \ldots, \widehat{\mu}_{c-1,t}, \frac{T_{c,t}\widehat{\mu}_{c,t}}{T_{c,t} + 1}, \widehat{\mu}_{c+1,t}, \ldots, \widehat{\mu}_{\mathscr{K},t}\right)^T.$$

Knowing that the probability for the next sample taken in subset c to be 1 is μ_c, we have the following expected decrease in the risk incurred by sampling a new instance in subset c' that we note

$$\begin{aligned} &\Delta_{c'} R_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \mu_c\right) \\ &\quad = \mu_{c'}\left(R_c\left(\mathbf{T}_t^c, \widehat{\boldsymbol{\mu}}_t^{c'+}, \mu_c\right) - R_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \mu_c\right)\right) \\ &\qquad + \left(1 - \mu_{c'}\right)\left(R_c\left(\mathbf{T}_t^c, \widehat{\boldsymbol{\mu}}_t^{c'-}, \mu_c\right) - R_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \mu_c\right)\right). \end{aligned} \quad (45)$$

With the global risk being

$$R\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \boldsymbol{\mu}\right) = \sum_{c=1}^{\mathscr{K}} R_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \mu_c\right), \quad (46)$$

its expected decrease relatively to a new sample in subset c is

$$\Delta_{c'} R\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \boldsymbol{\mu}\right) = \sum_{c=1}^{\mathscr{K}} \Delta_{c'} R_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \mu_c\right). \quad (47)$$

We then make a myopic minimization of the risk. Thus, at time t, our full knowledge algorithm samples the subset

$$c_t = \arg\min_c C_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \boldsymbol{\mu}\right), \quad (48)$$

where

$$C_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \boldsymbol{\mu}\right) = \Delta_c R\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \boldsymbol{\mu}\right) \quad (49)$$

is the criterion for the online allocation of the budget n.

Through this section, we have seen an online criterion, based on the maximum expected decrease in terms of risk, which a full knowledge algorithm would use to sample. The hypothesis made about the knowledge of the μ_c values is unrealistic because if they were known, the classification would be obvious; however, it allowed us to determine a good allocation strategy that our partial knowledge algorithm tries to attain. In the next section, we remove this hypothesis and use an optimistic approach to estimate the μ_c and at the same time allocate samples as close as possible to the full knowledge allocation.

5.3. Included Optimism. In this section, we introduce two optimistic algorithms based on the full knowledge criterion of the previous section. The values of μ_c are not given, so the selection strategy cannot use them directly. Although the labels acquired during the sampling process allow us to estimate them. The simple replacement of the true values with their estimates is unjustified and could lead to a bad allocation.

Instead, we should consider the exploration/exploitation tradeoff. More than the only estimates, we are able to compute a distribution on the belief of μ_c. A Bayesian Credible Interval, relatively to a probability δ, can thus be computed on the values of μ_c, as well as on the criterion.

Referring to the Optimism in the Face of Uncertainty approach, we define a selection strategy that samples at every time step the cluster for which the upper bound of the Bayesian Credible Interval is highest. This leads to an allocation of the samples as close as possible to the Full Knowledge allocation even though the true value of μ_c is not known.

Our first algorithm is OEMAL (Optimistic Error Minimization for Active Learning). It uses the set of partition as defined in the previous section. The classifier is only defined by the matrix P representing the influence of clusters on each other. The number of clusters in the thinnest partition is not limited. One particular case of OEMAL is when its clusters contain at most one instance from the pool. This makes P act as the covariance matrix used in kernel methods.

Our second algorithm is OEMAL-k (Optimistic Error Minimization for Active Learning: kernel version). It takes as input any covariance matrix and use it as the matrix P in OEMAL. This broadens the scope of the classifiers that can be used. Note that, in order to use Optimism in the Face of Uncertainty in a proper way, the matrix P still must not change. The rest of the section works for both algorithms.

Our algorithm is displayed in Algorithm 2. It takes one parameter δ which allows us to control the level of exploration used by our algorithm.

At any time, we are able to compute a Bayesian Credible Interval on the value of the parameters of each cluster. Each subset c of the thinnest partition is associated with a Bernoulli distribution of parameter μ_c. Thus, the estimated means are drawn from Binomial distributions. Beta distributions provide a family of conjugate prior probability distributions for Binomial distributions. With the prior probability distribution being Beta(1, 1) (uniform distribution), by Bayesian inference,

$$\mathbb{P}\left(\mu_c \mid \widehat{\mu}_c\right) = \text{Beta}\left(T_c\widehat{\mu}_c + 1, T_c\left(1 - \widehat{\mu}_c\right) + 1\right). \tag{50}$$

However, this inference does not consider observations from the neighbor clusters. Let us define $\mathbf{m} = (m_1, \ldots, m_{\mathscr{K}})^T$ with $\forall c \in [\![1, \mathscr{K}]\!]$,

$$m_c = \frac{\sum_{c'=1}^{\mathscr{K}} P_{c,c'}\mu_{c'} - 0.5}{\sum_{c'=1}^{\mathscr{K}} P_{c,c'}\mathbb{1}_{T_c>0}} + 0.5, \tag{51}$$

which is the decision criterion of the classifier defined in (39). In order to get an early guess about the value of $\boldsymbol{\mu}$, we chose to infer $\mathbf{m}$ instead, which aims to approach $\boldsymbol{\mu}$, and use it in place of $\boldsymbol{\mu}$. Even though $\mathbf{m} \neq \boldsymbol{\mu}$ and we may lose some accuracy, this is necessary for our algorithm to work well.

Input: δ, P
Initialize: Sample one instance at random
for $t = 1, \ldots, n$ **do**
  Compute μ_c^N and s_c^N from (52) and (53)
  Use one **Method 1** or **Method 2** to compute $\mathbb{P}(C_{c,t} = Q \mid \widehat{\boldsymbol{\mu}}_t, \mathbf{T}_t)$
  Compute $e_{k,t}$ from (61) Sample the cluster $\mathscr{X}_{c_t}$ with $c_t = \text{argmax}_c e_{c,t}$
end
Output: $l_{c,n}$ for all arms $1 \leq c \leq \mathscr{K}$

Algorithm 2: OEMAL/OEMAL-k.

First, let us state that $\widehat{\mathbf{m}}$ results from a sum of independent Binomial distributions; therefore assuming that the number of nonnull elements in each row of P is large enough, we can approximate that $\widehat{\mathbf{m}}$ follows a normal distribution.

The normal distribution provides a family of conjugate prior probability distributions for the normal distributions; thus, our belief about $\mathbf{m}$ follows a normal distribution with a mean of $\boldsymbol{\mu}^N = (\mu_1^N, \ldots, \mu_{\mathscr{K}}^N)^T$ with $\forall c \in \{1, \ldots, \mathscr{K}\}$,

$$\mu_c^N = \frac{\sum_{c'=1}^{\mathscr{K}} P_{c,c'}\widehat{\mu}_{c',t} - 0.5}{\sum_{c'=1}^{\mathscr{K}} P_{c,c'}\mathbb{1}_{T_c>0}} + 0.5 \tag{52}$$

and a standard deviation $\mathbf{s}^N = (s_1^N, \ldots, s_{\mathscr{K}}^N)^T$ with $\forall c \in \{1, \ldots, \mathscr{K}\}$,

$$s_c^N = \sqrt{\sum_{c'=1}^{\mathscr{K}} \left(\frac{P_{c,c'}}{\sum_{c'=1}^{\mathscr{K}} P_{c,c'}\mathbb{1}_{T_c>0}}\right)^2 \nu_{c'}^B} \tag{53}$$

with $\nu_{c'}^B = (T_{c'}^2\widehat{\mu}_{c'}(1 - \widehat{\mu}_{c'}) + T_{c'} + 1)/(T_{c'} + 2)^2(T_{c'} + 3)$, the variance of the Beta distribution.

Then,

$$\mathbb{P}\left(\mathbf{m} \mid \widehat{\boldsymbol{\mu}}\right) = \mathscr{N}\left(\boldsymbol{\mu}^N\left(\widehat{\boldsymbol{\mu}}, \mathbf{T}\right), \mathbf{s}^N\left(\widehat{\boldsymbol{\mu}}, \mathbf{T}\right)\right). \tag{54}$$

This algorithm, simulates the sampling of a new instance in order to estimate the resulting gain in risk. Within each simulation the distribution of the belief is updated by taking into account the new sample. Then, this new distribution of the belief is used to compute a distribution on the gain in risk.

Let us note y_s is the label of the instance sampled in the simulation and

$$\widehat{\boldsymbol{\mu}}_t^c = \left(\widehat{\mu}_{1,t}, \ldots, \widehat{\mu}_{c-1,t}, \frac{T_{c,t}\widehat{\mu}_{c,t} + y_s}{T_{c,t} + 1}, \widehat{\mu}_{c+1,t}, \ldots, \widehat{\mu}_{\mathscr{K},t}\right)^T. \tag{55}$$

We thus replace

$$\mathbb{P}\left(C_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \boldsymbol{\mu}\right) = Q \mid \widehat{\boldsymbol{\mu}}_t\right) \tag{56}$$

by

$$\begin{aligned}&\mathbb{P}\left(y_s \mid \mu_c = m_c\right)\mathbb{P}\left(m_c \mid \widehat{\boldsymbol{\mu}}_t\right)\\&\quad\cdot \mathbb{P}\left(C_c\left(\mathbf{T}_t, \widehat{\boldsymbol{\mu}}_t, \mathbf{m}\right) = Q \mid \widehat{\boldsymbol{\mu}}_t^c\right)\\&\quad= \sum_{\sum_{c'} q_{c'}=Q} \mathbb{P}\left(y_s \mid m_c\right)\mathbb{P}\left(m_c \mid \widehat{\boldsymbol{\mu}}_t\right)\\&\quad\cdot \prod_{c'} \mathbb{P}\left(\Delta_c R_{c'}\left(m_{c'}\right) = q_{c'} \mid \widehat{\boldsymbol{\mu}}_t^c\right).\end{aligned} \tag{57}$$

In order to compute this value we define two methods.

Method 1 (only for OEMAL). (i) Draw a high number of instantiations of the belief from $\mathbb{P}(\mathbf{m} \mid \widehat{\boldsymbol{\mu}})$.

(ii) Compute for each case the resulting value of the criterion.

(iii) Estimate the distribution of the criterion.

In our experiments, the set of partitions $\mathbf{N}$ we used is such that every cluster of the thinnest partition contained at most one instance from the pool. Thus, the value of μ_c is contained in $\{0, 1\}$.

Method 2 (if $\forall c \in [\![1, \mathcal{K}]\!] W_c = 1$). Compute the probability of μ_c being 0 as follows:

$$\mathbb{P}\left(\mu_c = 0 \mid \widehat{\boldsymbol{\mu}}_t\right) = \Phi\left(0.5, \boldsymbol{\mu}^N\left(\widehat{\boldsymbol{\mu}}_t, \mathbf{T}_t\right), \mathbf{s}^N\left(\widehat{\boldsymbol{\mu}}_t, \mathbf{T}_t\right)\right), \tag{58}$$

with Φ being the cumulative distribution function of the normal distribution:

$$\begin{aligned}&\sum_{\sum_{c'} q_{c'}=Q} \mathbb{P}\left(y_s \mid \mu_c\right)\mathbb{P}\left(\mu_c \mid \widehat{\boldsymbol{\mu}}_t\right)\\&\quad\cdot \prod_{c'} \mathbb{P}\left(\Delta_c R_{c'}\left(\mu_{c'}\right) = q_{c'} \mid \widehat{\boldsymbol{\mu}}_t^c\right)\end{aligned} \tag{59}$$

with

$$\Delta_c R_{c'} = \mathbb{1}_{\mu_{c'} \neq l^c_{c',t}} - \mathbb{1}_{\mu_{c'} \neq l_{c',t}} \tag{60}$$

and $l^c_{c',t}(\widehat{\boldsymbol{\mu}}_t^c)$ and $l_{c',t}(\widehat{\boldsymbol{\mu}}_t)$ from (39).

The simulation of a sample can lead to two states of the classifier. Depending on this state, the true risk is the sum of the true risk in each cluster, with at most two cases each ($q_{c'} \in \{-1, 1\}$ or $q_{c'} = 0$), depending on the value of μ_c in each cluster. We can then compute the probability of a value of the criterion by combining the probability of each case. As the difference of the true risk for one cluster is 0 whenever the prediction of the classifier does not change, we only consider clusters for which the prediction of the classifier changes. With this, the computation of the distribution of the criterion can be done in a reasonable time.

Having computed the distribution of the criterion, we can compute the upper bound of the Bayesian Credible Interval, which is

$$e_c = \arg\min_{\epsilon} \mathbb{P}\left(C_{c,t} > \epsilon \mid \widehat{\boldsymbol{\mu}}_t, \mathbf{T}_t\right) \leq \delta. \tag{61}$$

In this section, we derived two optimistic algorithms that address the active learning problem for classification. OEMAL considers a set of partitions of the instance space as well as the thinnest partition, resulting from overlapping this set. OEMAL-k considers a kernel. They both find the cluster of the thinnest partition for which sampling in it results in the greatest decrease of the true risk.

5.4. Computational Complexity. Let $\mathcal{K}$ be the number of clusters of the thinnest partition, and let $\mathcal{K}_U$ be the number of clusters with at least one unlabeled instance. The computation of the criterion requires $\mathcal{O}(\mathcal{K}^2)$ time complexity.

Using Method 1, let n_{its} be the number of instantiations of the belief. The selection of the next cluster thus requires $\mathcal{O}(\mathcal{K}^2\mathcal{K}_U n_{\text{its}})$ time complexity. Indeed, at each time step, it computes the criterion for n_{its} values of the parameters drawn from the posterior and for the simulation of sample in each one of $\mathcal{K}_U$ clusters.

In the case where each cluster contains only one instance from the pool, the decrease in risk in one cluster can be 0 if the predicted label does not change, or 1 or −1 depending on the true label, if it changes. Let n_{changes} be the number of clusters seeing its label change for the considered simulation of sample.

Using Method 2, the computational complexity of the combinatorial procedure is $\mathcal{O}(n^2_{\text{changes}})$. Thus, the selection of the next cluster requires a computational complexity of $\mathcal{O}(\mathcal{K}^2\mathcal{K}_U n^2_{\text{changes}})$.

Added to the fact that Method 2 is more precise than Method 1 because the computed cumulative distribution is exact, it is also faster, because n_{changes} rapidly decreases while acquiring more samples.

Note that the number of partitions in the set is not involved in the computational complexity. Indeed, they are only involved in the computation of the relations between subsets of the thinnest partition which is made beforehand. This is another advantage compared with the use of random forests, where each tree has to be recomputed at every time step.

6. Evaluation

In this section, we evaluate the algorithms introduced in the previous sections. Each of the evaluation will be the scope of a comparison between our algorithms and state-of-the art algorithms. Algorithms are evaluated on two different benchmarks depending on the classifier used.

In the case of independent clusters, either the instance space is already discrete, or it is continuous and must be partitioned. In the latter, the partition has to be chosen before taking any samples and cannot be changed after that. The predicted label in each cluster depends only on samples drawn in it. Thus, the number of clusters is limited by the budget of samples and by a desired good learning rate. This makes the partition rough and the classifier noncompetitive on real world datasets. However, the small number of parameters needed to represent any real world problem allows us to build a representative benchmark.

In the case of information sharing clusters, a partition is also given beforehand but the predicted label in each cluster may depend on all the samples. The number of clusters is thus not limited as before, and an extremely refined partition, where each cluster contains only one instance from the pool, may be used. Thus, this algorithm may be evaluated on real world problems with a continuous instance space.

6.1. Independent Clusters

6.1.1. Practical Implementation and Experimental Setup. Two optimistic algorithms were introduced in the context of independent clusters. Each one based on different full knowledge criteria: the first one defines a one-shot allocation of the samples, which means that the optimal number of samples to take in each cluster is only function of its parameter and of the budget, and the second one defines a fully sequential allocation of the samples, which means that the optimal number of points to take in each cluster depends also on the samples drawn.

The performance of the algorithms closely depends on the choice of the partition. Clusters may regroup instances with a great dispersion of labels, either because the instances are subject to a lot of noise, or because the mean label varies rapidly. In this second case, a better discretization of the instance space causes a better true risk when the correct label is given. But this implies increasing the number of clusters, and for the prediction of the label to be equivalently accurate, we need a higher budget. As the problem of the choice of the partition is not studied here, the use of our algorithm on real world datasets with a continuous instance space would not be competitive with methods designed for this problem. Instead, we show that, given any partition, the algorithm performs better than other algorithms with the same constraints.

In this problem, where the classifier predicts one label per cluster, the location in the cluster of the sampled instance is of no interest. Therefore, every cluster can be seen as a pool of instances returning labels with a certain proportion. With the labels being either 0 or 1, this leads to a representation of clusters as Bernoulli distribution only characterized by one parameter. The relative position of the clusters is also of no interest for the classifier. Consequently, every problem our algorithm could encounter is only characterized by

(i) the number of clusters K,

(ii) the parameter of the Bernoulli distribution associated with each cluster $(\mu_k)_{k\in[\![1,K]\!]}$,

(iii) the weight of each cluster $(w_k)_{k\in[\![1,K]\!]}$.

If two problems share the same characteristics, our algorithms will act the same on them.

Our first benchmark is thus to generate randomly a set of problems by drawing random parameters from the above definition. Then, the tested algorithms are launched on each problem of the set, and their true risk is recorded at every time step. The current predicted label for each cluster is compared to the correct one (the round value of μ_k), and the true risk is the weight sum of this logical comparison. Note that because the problems are built in, the true parameter is known and we do not need a test set. Likewise, the samples are directly drawn from the distributions and not drawn from a pool. The global performance of the algorithms results from averaging the true risk at each time step. In our experiments, the benchmark contains 1000 problems generated with

(i) K drawn uniformly in $[\![1, 50]\!]$,

(ii) $(\mu_k)_{k\in[\![1,K]\!]}$ drawn uniformly in $[0, 1]^K$,

(iii) $(w_k)_{k\in[\![1,K]\!]}$ drawn uniformly in $[0, 1]^K$.

The weights are normalized, but this is just anecdotal as it does not affect the behavior of the algorithms. We use a budget of 1000 samples. The results are displayed with the time step range starting at 100 in order to be able to differentiate algorithms.

Our second benchmark comes from the fact that not all state-of-the-art algorithms consider the weight given to clusters. In order to make sure that the good performance of our algorithms is not only due to this consideration, in this benchmark the weights are equal for all clusters. The problem of allocation consists in defining which cluster has priority against another. Thus, any problem is a subproblem of the one containing an infinite number of clusters containing all the values possible for the parameter. Then, we generate one problem containing many clusters with the widest variety of parameters. In our experiments, this problem is generated with

(i) $K = 100$,

(ii) $\forall k \in [\![1, K]\!], \mu_k = k/K$,

(iii) $\forall k \in [\![1, K]\!], w_k = 1/K$.

We run the algorithms 1000 times on this problem to face the randomness of the samples and average their true risk at every time step. The results are displayed with the time step range starting at 100.

To demonstrate the effectiveness of our algorithms, we compare them with existing state-of-the-art methods.

Random Sampling. This is the simplest baseline; at every time step the sampled cluster is drawn uniformly in $[\![1, K]\!]$. Any active learning algorithm should do better than that.

Uniform Sampling. This is another simple baseline, and the clusters are sampled uniformly. At every time step the sampled cluster is drawn uniformly among the least sampled ones.

Monte Carlo Upper Confidence Bound (MC-UCB) (see [20]). This algorithm also uses an optimistic approach, although it is not originally designed for classification. It aims to estimate uniformly well the parameter of the distribution in each cluster. We use this estimation to predict the label. This algorithm also considers a weight for each cluster.

EffECXtive (see [12]). This algorithm tries to identify the best hypothesis from a set by successively sampling the instance for which the expected reduction in weight is the greatest. The weight is defined as the sum of the squared probabilities of

hypotheses. In our case, a hypothesis is characterized by the label given to each cluster, and the probability of a hypothesis is

$$\mathbb{P}\left(h = (y_k)_{k\in[\![1,K]\!]}\right) = \prod_{k=1}^{K} \mathbb{P}\left([\mu_k] = y_k \mid \widehat{\mu}_k\right), \quad (62)$$

where $\mathbb{P}([\mu_k] = 1 \mid \widehat{\mu}_k) = \mathcal{I}_{0.5}(T_k\widehat{\mu}_k + 1, T_k(1 - \widehat{\mu}_k) + 1)$ with $\mathcal{I}_x(a, b)$ being the regularized incomplete beta function.

The three optimistic algorithms evaluated in this section, namely, MC-UCB, OALC-DI, and OALC-DD, use a parameter δ. This parameter has been tuned by using a grid search.

6.1.2. Evaluation. First, we evaluate the performance of the algorithms on Benchmark 1. The results of the evaluation are displayed on Figure 3(a).

We first compare MC-UCB with OALC-DI. MC-UCB is an optimistic algorithm which tries to estimate uniformly well the mean value of the distribution in each cluster. This can be related to active learning in the case of regression. A good estimate of the mean value leads to a good prediction for the label. Therefore, this algorithm may be used on a classification problem even though it may not be the best. Indeed, it will spend effort to be precise on the estimation of the mean value even if this precision is of no use for the decision of the label. Those efforts could have been spent in a different cluster where the label uncertainty is larger. OALC-DI is the closest of our algorithms to MC-UCB as they both consider a Full Knowledge criterion that gives the optimal allocation of the budget without knowing the results of the samples. The two differ in that OALC-DI is specifically designed for classification. We can see that OALC-DI shows a significant improvement over MC-UCB, which indicates that working with an algorithm adapted for classification is not to be neglected.

We then compare OALC-DI and OALC-DD. Those two algorithms aim both to minimize the same objective function, which is the expected true risk of the classifier. Note that using directly the true risk based on the binary loss is usually avoided by minimizing a convex proxy of it. Although the second one is based on a full knowledge criterion that takes the current state of the classifier, depending on the results of the samples so far, into account whereas the first one is not. We can see that, as expected, the version of our algorithm based on a data-dependent full knowledge criterion behaves better than the one based on a data-independent one. Note that even though the difference in performance is only of 0.0016 at time step 1000, which means that 0.16% of the instances will be classified better, the number of time steps required to attain the performance of OALC-DI at time step 1000 is of 625 for OALC-DD, which is a save of 37.5% of the labeled instances.

Finally, let us take a look at the performances of OALC-DD and EffECXtive. Both algorithms are very similar as they greedily minimize an expected objective function. The difference is that EffECXtive uses a proxy of the true risk, the Rényi entropy, whereas OALC-DD, by allowing it to depend on the true parameters of the distribution before being optimist regarding it, is closer to the actual true risk. Note that EffECXtive, in our adaptation to a partition of the instance space, does not take into account relative importance given to clusters (weights), which are not trivial to include. The results show that OALC-DD performs slightly better than EffECXtive.

Let us now evaluate the performance on Benchmark 2. The results are displayed in Figure 3(b). This problem contains more clusters than any other problem in Benchmark 1. A higher budget is thus needed to attain comparable performances. Still, the range of time steps does not change. This allows us to focus on the first phase of the algorithms. In this problem, the weights are equal for all clusters, so that the quality of an algorithm is not only based on the fact it has this feature. In this problem, the best true risk is equal to 0.25.

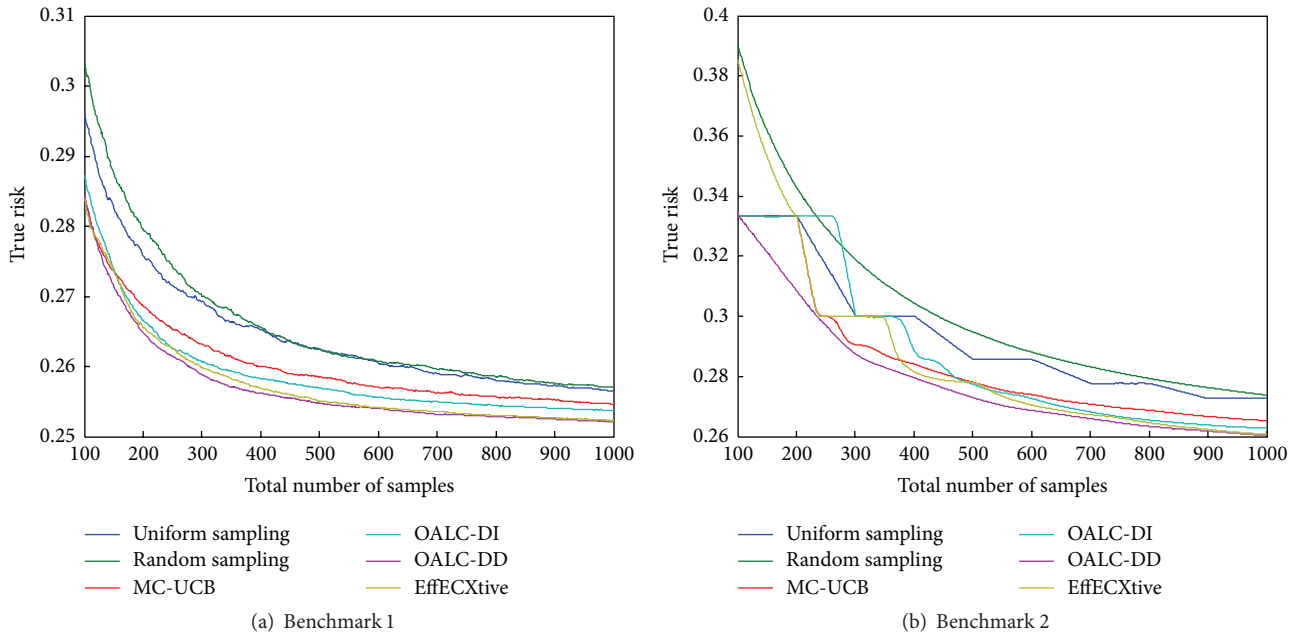
Note that this range starts at 100 labeled examples. But since most algorithms need every cluster to be sampled at least once, at time step 100, each one of the 100 clusters will be sampled once, and this is for every algorithm, apart from random sampling. This is why every algorithm performance starts at the same level. We thus do not lose much by starting at this time step. The results of the first sample are 0 or 1 leading to the same precision on the prediction, and some algorithms need every cluster to be sampled twice, and this is why the performance at time step 200 is the same for most algorithms. EffECXtive performance at 100 time step is not the same as others as it samples indifferently clusters with no samples and clusters with one sample. But it has the same performance as others at time step 200. OALC-DD prefers to sample clusters that have received two samples 0 and 1 than one sample, which appears as a good behavior relatively to the results.

We can see that OALC-DD has better performance than all other algorithms at every time step.

6.2. Information Sharing Clusters

6.2.1. Practical Implementation and Experimental Setup. Two other optimistic algorithms were introduced. Instead of one partition, OEMAL uses a set of partitions which all compute the estimates independently and merge to predict the final label. The thinnest partition was defined, which is the partition resulting from the intersection of all the partitions in the set. We have seen that we could use a representation of the classifier which involves only the clusters of the thinnest partition and a matrix P which tells how much the estimate in one cluster plays a role in predicting the label in another cluster. Using a set of partitions is thus equivalent to one partition with clusters that share information.

The classifier used by our algorithm shares some resemblance with Random Forests [22] as it uses a set of partitions and average of the prediction criterion of each one of them. In our algorithm, the set of partitions has to remain the same throughout the progress. Whereas in Random Forests, they are recomputed at each step. Hence, it cannot adapt to the received labels and must be defined at the beginning. We use purely random partitions of the instance space that are not based on the instances in the pool. It is thus more closely related to tile coding which is a function approximation method commonly used in the field of Reinforcement Learning [23].



(a) Benchmark 1 (b) Benchmark 2

Figure 3: Evaluation of algorithms in the case of independent clusters.

The nature of the partitions in the set and its number was not defined in the algorithm. In fact, the algorithm works given any set of partitions. The performances of the classifier clearly depend on the partitions in the set; for example, if all the partitions in the set are the same, then the problem is reduced to the one partition problem. But, as long as partitions are diverse, their shape is not as determinant as before. In the context of one partition, the number of clusters in it could not be too large because the belief on the parameter which guided the selection strategy was only based on observations in its cluster. Now, the number of clusters in the thinnest partition has no limitation. This allows us to work with a continuous instance space without the loss in performance incurred by the choice of the partition.

The partitions we use consist of 7 successive random splits of the instance space along random dimensions. The number of partitions in the set is 10.000. At the end, the clusters of the thinnest partition contain either 1 or no instance from the pool.

OEMAL-k replaces the matrix P in OEMAL by a covariance matrix relatively to a kernel. In the experiments, we use a Gaussian kernel covariance matrix:

$$P_{i,j} = e^{-|x_i - x_j|^2/2s^2}. \quad (63)$$

The scale parameter s has been tuned to give the best performance with a full training set.

We thus evaluate our algorithm on several real world datasets from the UCI Machine Learning Repository [24]. The four datasets used in this paper are Australian, Diabetes, Heart, and Wdbc. In all those datasets, the instances belong to a continuous instance space. In each run of experiments, the dataset is randomly divided into two. The first half is used as the pool of unlabeled instances in which the algorithm is allowed to pick, while the second half is used as the test set. At each time step, the true risk of the current prediction is estimated via the test set and recorded. The global performance of an algorithm at each time step is computed as the average of the performance among the runs. In our experiments, the number of runs is set to 1.000. The parameter δ is tuned for every dataset using a grid search.

We compare our algorithm with existing state-of-the-art method and some baselines.

Random Sampling. This is the simplest baseline. At each time step a random instance is drawn from the pool of unlabelled instances.

Full Knowledge. This is the best algorithm we can do. At each time step, an instance is selected according to the full knowledge criterion. The values of the parameters are used; thus, it is unrealistic but it serves to show how well the exploration/exploitation tradeoff is achieved.

Uncertainty Sampling (see [7]). This is the most common active learning algorithm. At each time step, the instance which is most uncertain about how to label is selected.

6.2.2. Evaluation. Figure 4 displays the results of the evaluation for the four following datasets: Figure 4(a) Australian, Figure 4(b) Diabetes, Figure 4(c) Heart, and Figure 4(d) Wdbc.

We built our optimistic algorithm by first defining a full knowledge criterion which guides the allocation of samples in the case where the true values of the parameters are known from the beginning. This is then used by the algorithm as a target allocation to attain in the case where those values are unknown. The performances of OEMAL are thus limited by those of the full knowledge allocation. We display the performances of the Full Knowledge allocation as a baseline

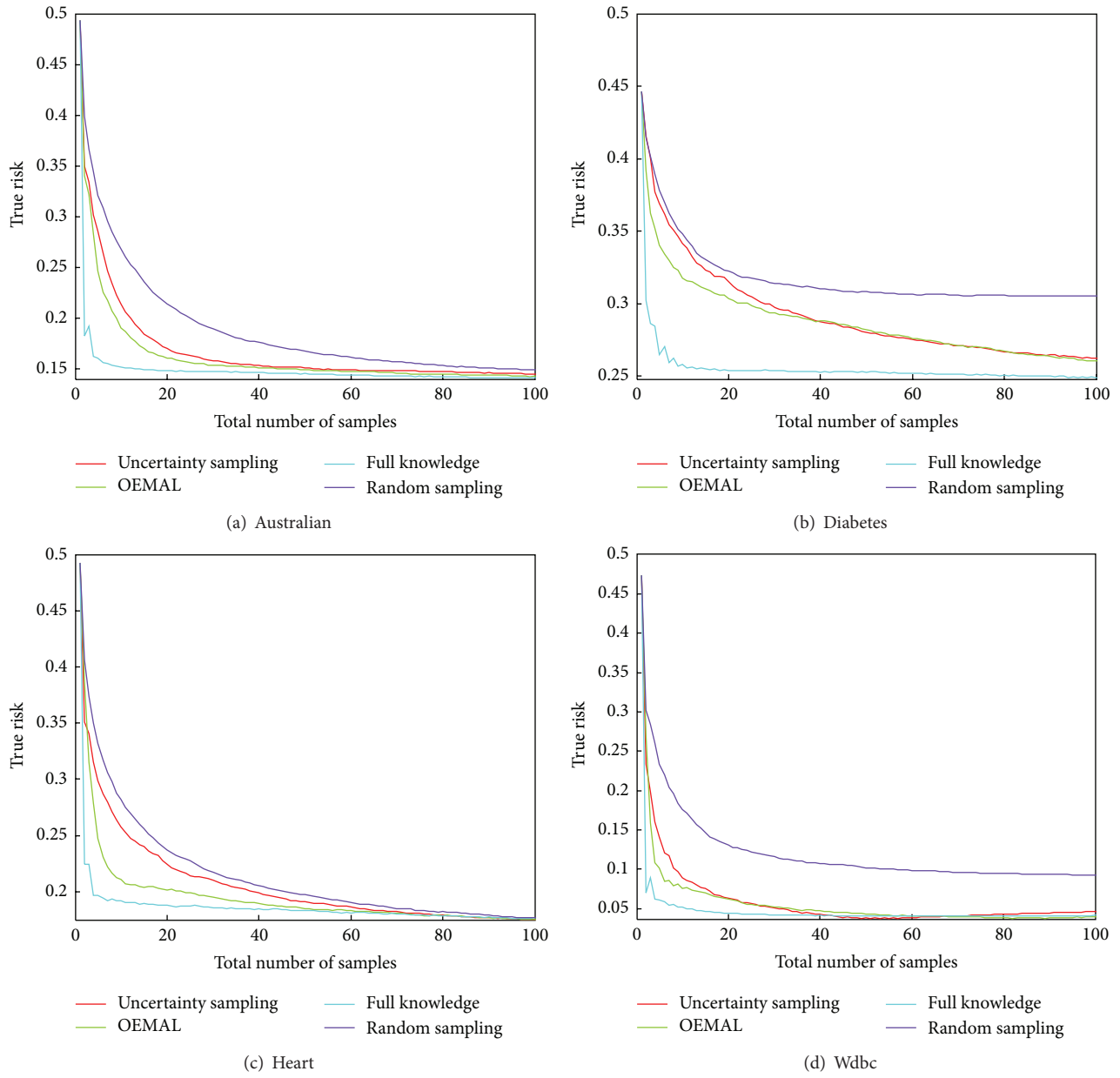


Figure 4: Evaluation of algorithms in the case of information sharing clusters.

for the Optimistic algorithm. If they both have the same performances, then the only way to improve the algorithm is to design a better Full Knowledge criterion. Otherwise, the way to manage uncertainty is to improve. In other words, it tells if the exploration/exploitation tradeoff is well achieved. The full knowledge allocation, for its part, sees its performances limited by the classifier, which may not have better performances given any set of instances of size given by the time step. Also, whereas in the dependent clusters case the full knowledge allocation was known to be optimal, meaning that we could not achieve better performances with a different allocation, this is not the case anymore. Indeed, the myopic minimization of the risk has no guaranty to lead to the optimal allocation. For example, the best performances of the classifier at time step 2 could be achieved by the inclusion of a pair of instances that does not contain the instance that leads to the best performance at time step 1. Although the empirical results show that using a myopic minimization in full knowledge performs quite well, we can see that on the Wdbc dataset Figure 4(d), for a short period around 55 labeled instances, uncertainty sampling achieves slightly better performance than the full knowledge allocation. The optimistic algorithm can not thus do better on this period. Still, we can see that it outperforms it on the 20 first samples as well as it keeps high performance on the end while Uncertainty Sampling seems to lose accuracy.

This last phenomenon can be explained. Let us look at Figure 5 where we display the results for the Wdbc

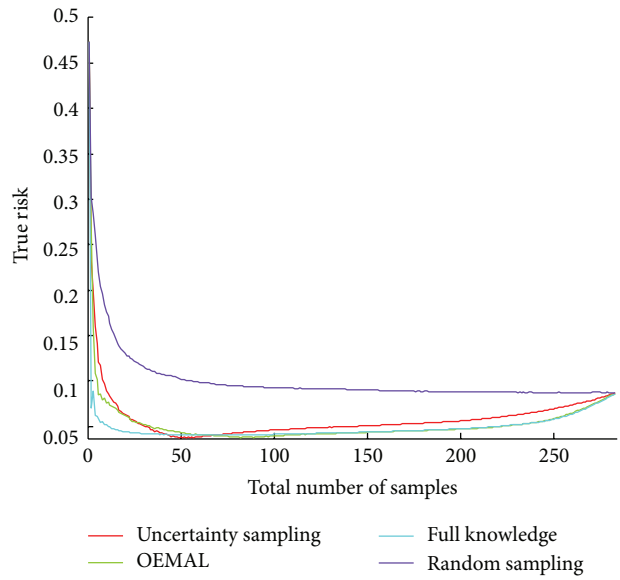


Figure 5: Evaluation of the algorithms on Wdbc until all the instances of the pool are sampled.

dataset with the range of time steps extended to show the behavior of the algorithms until the last sample of the pool is retrieved. We can see the performance of all the algorithm decreases while approaching the end. We may think that the performance can only increase with the number of samples taken into account by the classifier. We know that this is not necessarily true, particularly if the classifier overfits. Also, in active learning, one can select a subset of instances that achieves better performance than when taking all the instances from the pool. The question of a stopping criterion has already been studied in order to avoid spending useless resources. Here, we see that it is even more crucial because it could lead to better performances. In OEMAL, the criterion used represents the maximum decrease of the true risk one can expect by taking a particular sample. Thus, if the value of the criterion is less than 0, this means that there is a high probability $(1 - \delta)$ that the true risk will not decrease but increase. In this case, it is preferable not to sample this instance. But if the maximum criterion is less than 0, it is preferable not to sample at all. We use this as a stopping criterion. We compare the final performances of our algorithm with and without the stopping criterion for different datasets in Table 1.

Two of the four datasets see their true risk increase at the end, Diabetes and Wdbc. We can see that the use of a stopping criterion is efficient in those cases and does not greatly alter the performances in other cases.

The matrix P appearing in the classifier we used so far is derived from the use of a set of partitions. The value in each cell of the matrix corresponds to the weight of each estimation in each prediction. We saw that it was possible to use clusters of the thinnest partition that contain only one instance from the pool. Instances are linked to others through P. Inherently, in this case P fully specifies the data manifold structure. The theory that leads to OEMAL is built upon this set of partition,

Table 1: Final true risk of the classifier using OEMAL with or without using a stopping criterion.

Dataset	OEMAL with stopping criterion	OEMAL without stopping criterion
Australian	0.1367	0.1382
Diabetes	0.2422	0.3095
Heart	0.1724	0.1722
Wdbc	0.0397	0.0860

but we can imagine using a matrix P of any kind, specifying other weights between instances. OEMAL may still work in this context. Particularly, we can adapt this algorithm to the active learning of kernel classifiers. We thus use a Gaussian kernel covariance matrix for P. We now denote the kernel version of OEMAL by OEMAL-k. The results are displayed in Figure 6.

We can see that OEMAL-k always does better than uncertainty sampling. Even in the Wdbc dataset where uncertainty sampling performs worse than random sampling, OEMAL-k manages to get better performance. One important thing in active learning is the choice of the classifier. With a well-fitted classifier, even the random sampling strategy could perform better than the best active learning strategy on a poor classifier. It is thus convenient that our algorithm is not limited only to one kind of classifier and can easily generalize to kernel classifiers. For example, on the Wdbc dataset Figures 4(d) and 6(d), the random sampling strategy performs significantly better when using a kernel than a set of partitions, and OEMAL-k keeps improving the results. On the other hand, on the Diabetes dataset Figure 6(b), the random sampling strategy is also better with the kernel classifier, but OEMAL performs better than OEMAL-k. As we have seen in Figure 5, the active learning strategy of the best performance for the classifier may be achieved with only a subset of instances. Maybe the former classifier had a better potential than the last. This is confirmed by the performance of the full knowledge criterion.

In this section, we evaluated the performance of OEMAL which is built on this approach on several real world datasets. Thus, we demonstrated that the *Optimism in the Face of Uncertainty* approach can be used for active learning in classification. We saw that it performed comparatively well to a famous state-of-the art algorithm. We also evaluated OEMAL-k and showed that our algorithm could be generalized to kernel methods or any graph based method where the instances are linked to otherr by a weight matrix P.

7. Conclusion

In this paper, we show that the problem of active learning in classification can be studied through the eye of Optimism in the Face of Uncertainty. This has the advantage to allow the selection criterion to be defined as close as possible to the evaluation function. It introduces three error minimization algorithms which use this approach. The experiments, conducted on built-in problems as well as real world datasets,

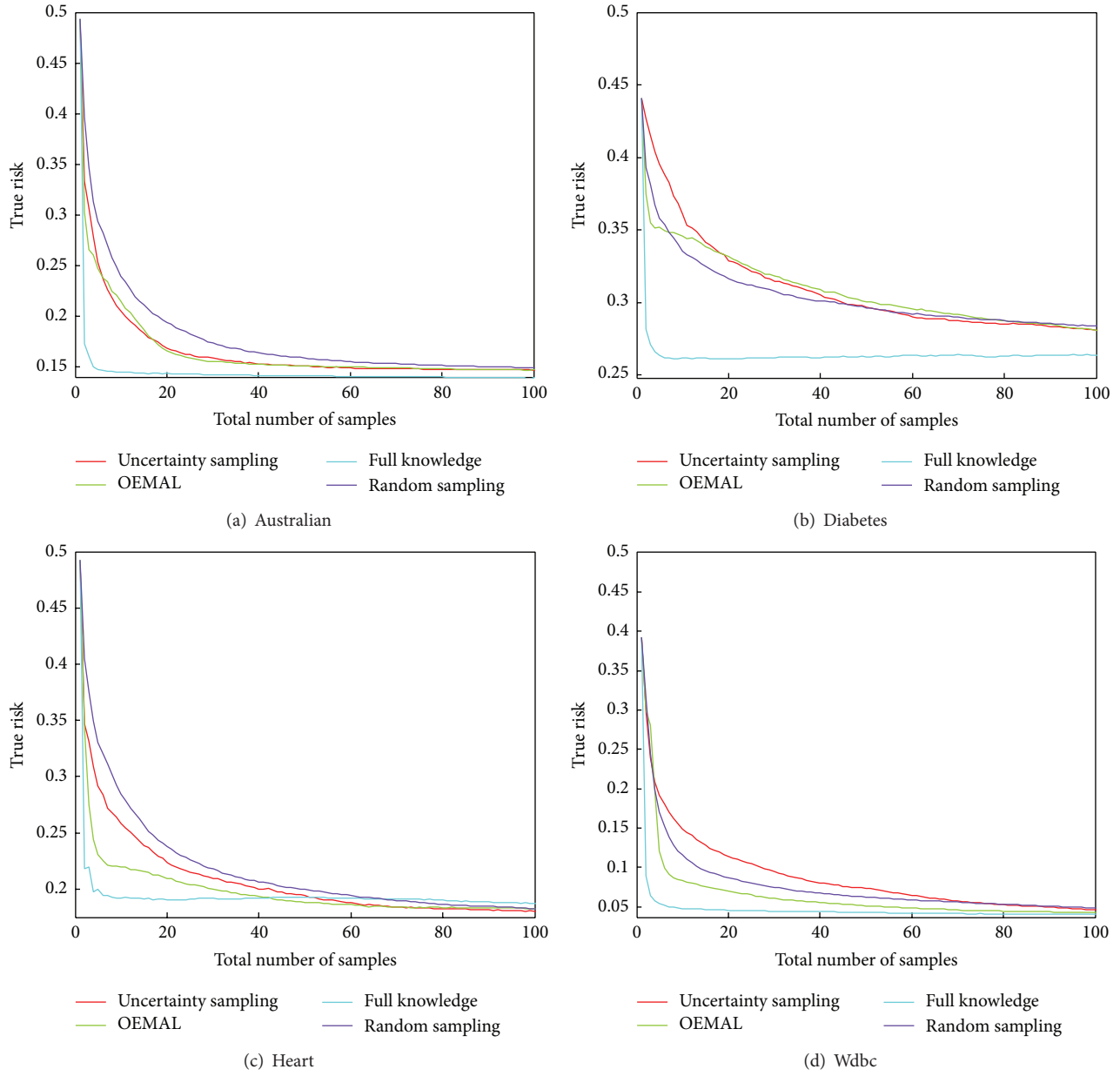


Figure 6: Evaluation of algorithms in the case of information sharing clusters.

show that they perform comparatively well to state-of-the-art methods. An extension of the last algorithm shows that it can be generalized to other kernels. We however constrained the matrix P to remain the same all along the progress of the algorithm. Our perspective is to work with a changing matrix P such as in kernel regression or Gaussian processes where the variance of the estimates is given.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] D. Cohn, L. Atlas, and R. Ladner, "Improving generalization with active learning," *Machine Learning*, vol. 15, no. 2, pp. 201–221, 1994.

[2] M. H. DeGroot, *Optimal Statistical Decisions*, vol. 82, John Wiley & Sons, Hoboken, NJ, USA, 2005.

[3] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3-4, pp. 285–294, 1933.

[4] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *Foundations and Trends in Machine Learning*, vol. 5, no. 1, pp. 1–122, 2012.

[5] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2-3, pp. 235–256, 2002.

[6] B. Settles, *Active Learning Literature Survey*, University of Wisconsin, Madison, Wis, USA, 2010.

[7] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *SIGIR '94: Proceedings of the Seventeenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, organised by Dublin City University*, pp. 3–12, Springer, New York, NY, USA, 1994.

[8] H. S. Seung, M. Opper, and H. Sompolinsky, "Query by committee," in *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pp. 287–294, ACM, July 1992.

[9] R. D. Nowak, "Noisy generalized binary search," in *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS '09)*, Vancouver, Canada, 2009.

[10] M.-F. Balcan, A. Beygelzimer, and J. Langford, "Agnostic active learning," in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, Pittsburgh, Pa, USA, June 2006.

[11] M. Naghshvar, T. Javidi, and K. Chaudhuri, "Noisy Bayesian active learning," in *Proceedings of the 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton '12)*, pp. 1626–1633, IEEE, October 2012.

[12] D. Golovin, A. Krause, and D. Ray, "Near-optimal bayesian active learning with noisy observations," in *Proceedings of the 24th Annual Conference on Neural Information Processing Systems (NIPS '10)*, Vancouver, Canada, December 2010.

[13] Y. Guo, "Active instance sampling via matrix partition," in *Advances in Neural Information Processing Systems*, pp. 802–810, 2010.

[14] B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP '08)*, pp. 1070–1079, Association for Computational Linguistics, 2008.

[15] N. Roy and A. McCallum, "Toward optimal active learning through Monte Carlo estimation of error reduction," in *Proceedings of the International Conference on Machine Learning (ICML '01)*, Williamstown, Mass, USA, 2001.

[16] X. Zhu, J. Lafferty, and Z. Ghahramani, "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of the Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining (ICML '03)*, pp. 58–65, Washington, DC, USA, 2003.

[17] A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell, "Active learning with Gaussian processes for object categorization," in *Proceedings of the 11th IEEE International Conference on Computer Vision (ICCV '07)*, pp. 1–8, IEEE, Rio de Janeiro, Brazil, October 2007.

[18] Q. Gu, T. Zhang, and J. Han, "Batchmode active learning via error bound minimization," in *Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI '14)*, vol. 51, pp. 300–309, Quebec City, Canada, 2014.

[19] R. Ganti and A. G. Gray, "Building bridges: viewing active learning from the multi-armed bandit lens," in *Uncertainty in Artificial Intelligence*, p. 232, 2013.

[20] A. Carpentier and R. Munos, "Finite time analysis of stratified sampling for Monte Carlo," in *Proceedings of the Annual Conference on Neural Information Processing Systems*, 2011.

[21] E. T. Jaynes and O. Kempthorne, "Confidence intervals vs bayesian intervals," in *Foundations of Probability Theory, Statistical Inference, and Statistical Theories of Science*, pp. 175–257, Springer, 1976.

[22] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[23] A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1998.

[24] M. Lichman, "UCI machine learning repository," 2013.