Querying Partially Labelled Data to Improve a K-nn Classifier

Vu-Linh Nguyen, Sébastien Destercke, Marie-Hélène Masson
UMR CNRS 7253 Heudiasyc, Sorbonne Université,
Université de Technologie de Compiègne CS 60319 - 60203 Compiègne cedex, France
{linh.nguyen, sebastien.destercke, mylene.masson}@hds.utc.fr

Abstract

When learning from instances whose output labels may be partial, the problem of knowing which of these output labels should be made precise to improve the accuracy of predictions arises. This problem can be seen as the intersection of two tasks: the one of learning from partial labels and the one of active learning, where the goal is to provide the labels of additional instances to improve the model accuracy. In this paper, we propose querying strategies of partial labels for the well-known K-nn classifier. We propose different criteria of increasing complexity, using among other things the amount of ambiguity that partial labels introduce in the K-nn decision rule. We then show that our strategies usually outperform simple baseline schemes, and that more complex strategies provide a faster improvement of the model accuracies.

Introduction

Recently, the problem of learning models from imprecise output labels, where we only know the true label to belong to some set, has gained some interest. Such labels appear for example in image labeling (Cour, Sapp, and Taskar 2011) or language processing (Yang and Vozila 2014). In such problems, partial labels may come from easy to obtain high-level information (for instance, character names in subtitles to identify those characters present in an image/video). The problem of learning from partial labels is known under various names: "learning with partial labels" (Cour, Sapp, and Taskar 2011), "learning with ambiguous labels" (Hüllermeier and Beringer 2006) or "superset label learning" (Liu and Dietterich 2014). In these works, authors have either proposed general schemes to learn from partial labels, for instance by adapting the loss function to partial labels (Cour, Sapp, and Taskar 2011), or to adapt specific algorithms (such as K-nn or decision trees) to the case of partial labels (Hüllermeier and Beringer 2006).

In general, the less partial are the labels, the better these techniques will perform. In the spirit of active learning techniques, this paper addresses the problem of finding which partial labels should be disambiguated by an oracle (expert) in order to achieve better performances. In active learning, one classically starts from a set of labelled data, and queries within a pool of data with missing labels. The case of partial labels shares similarities with this task, but also has some important differences. First, the information provided by partial labels should be used in the learning process to determine which one should be queried. Second, in active learning the default assumption that labels are missing-at-random (MAR) is reasonable, which is not necessarily the case with partial labels. For example, the partiality of the labels may be due to some specific values of features, or to the fact that some situations are more ambiguous than others for the labeller. A first possible solution is then to make some assumptions about the incompleteness process generating partial labels (Cour, Sapp, and Taskar 2011), which may be difficult to check in practice. A second approach (adopted in this paper) is to adopt a robust view consisting in considering all possible replacements of the partial labels.

More specifically, we look at the popular K-nn classifiers (Cover and Hart 1967; Wu et al. 2008), for which different algorithms handling partial labels already exist (Hüllermeier and Beringer 2006; Zhang and Yu 2015). In order to find which instance to query, we propose in Section a general scheme based on measuring the potential impact of knowing the true label of a given instance. We then propose specific measures to assess this impact. The first one, explained in Section , is simply based on the number of decisions in which the instance participates, while the other ones explained in Section also consider whether a partial label introduces some ambiguity in the decision, using some notions issued from social choice theory (Moulin et al. 2016) to do so. Finally, some experiments in Section show the effectiveness of our proposals.

General Setting and Querying Schema

This section introduces the basic notations used in this paper as well as our general querying scheme.

Setting

In our setting, we assume that we have a training set $\mathbf{D} = \{(\mathbf{x}_n, \mathbf{y}_n) | n = 1, \ldots, N\}$ used to make predictions, with $\mathbf{x}_n \in \mathbb{R}^P$ the features and $\mathbf{y}_n \subseteq \Omega = \{\lambda_1, \ldots, \lambda_M\}$ potentially imprecise labels. As usual when working with partial labels (Cour et al. 2009; Cour, Sapp, and Taskar 2011; Yang and Vozila 2014), we assume that $\mathbf{y}_n$ contains the true label. We also assume that we have one unlabelled target set

Figure 1: Example: 3-nn classifiers.

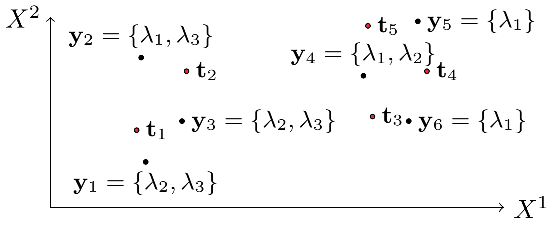
X^2
$\mathbf{y}_2 = \{\lambda_1, \lambda_3\}$
$\mathbf{t}_2$
$\mathbf{t}_5$
$\mathbf{y}_5 = \{\lambda_1\}$
$\mathbf{y}_4 = \{\lambda_1, \lambda_2\}$
$\mathbf{t}_4$
$\mathbf{t}_1$
$\mathbf{y}_3 = \{\lambda_2, \lambda_3\}$
$\mathbf{t}_3$
$\mathbf{y}_6 = \{\lambda_1\}$
$\mathbf{y}_1 = \{\lambda_2, \lambda_3\}$
X^1

Table 1: Weights and neighbours of Example 1

$\mathbf{t}$	$\mathbf{N_t}$	$\mathbf{w_t}$
$\mathbf{t}_1$	$\{\mathbf{x}_1, \mathbf{x}_3, \mathbf{x}_2\}$	(0.9,0.8,0.7)
$\mathbf{t}_2$	$\{\mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_1\}$	(0.8,0.8,0.4)
$\mathbf{t}_3$	$\{\mathbf{x}_6, \mathbf{x}_4, \mathbf{x}_5\}$	(0.8,0.8,0.4)
$\mathbf{t}_4$	$\{\mathbf{x}_6, \mathbf{x}_4, \mathbf{x}_5\}$	(0.7,0.7,0.7)
$\mathbf{t}_5$	$\{\mathbf{x}_4, \mathbf{x}_5, \mathbf{x}_6\}$	(0.8,0.8,0.4)

$\mathbf{T} = \{\mathbf{t}_j | j = 1, \dots, T\}$ that will be used to determine the partial labels to query and can be defined differently based on the usage purposes as point out latter in Section .

For a given instance $\mathbf{t}$ and a value K, its set of neighbours in $\mathbf{D}$ is denoted by $\mathbf{N_t} = \{\mathbf{x}_k^{\mathbf{t}} | k = 1, \dots, K\}$ where $\mathbf{x}_k^{\mathbf{t}}$ is its kth nearest neighbour. We will also say that $\mathbf{x}_n \in \mathbf{N_t}$ if $\mathbf{x}_n$ is among the K nearest neighbours of a given instance $\mathbf{t}$. We also assume that we have a vector $\mathbf{w_t} = (w_1^{\mathbf{t}}, \dots, w_K^{\mathbf{t}})$ weighting each neighbour in $\mathbf{N_t}$ according to its distance to the target. Similarly, for a training instance $\mathbf{x}_n \in \mathbf{D}$, we denote by $\mathbf{G}_{\mathbf{x}_n} = \{\mathbf{t} | \mathbf{x}_n \in \mathbf{N_t}\}$ the set of target instances of which $\mathbf{x}_n$ is a nearest neighbour.

In the remainder of this paper, we will need a way to take a decision in presence of partial labels. In this paper, we will consider the simple yet efficient method (Hüllermeier and Beringer 2006) that consists in predicting

$$h(\mathbf{t}) = \arg\max_{\lambda \in \Omega} \sum_{\mathbf{x}_k^{\mathbf{t}} \in \mathbf{N_t}} w_k^{\mathbf{t}} \mathbb{1}_{\lambda \in \mathbf{y}_k^{\mathbf{t}}}, \tag{1}$$

with $\mathbb{1}_A$ the indicator function of A ($\mathbb{1}_A = 1$ if A is true and 0 otherwise). The idea of the above method is to count one (weighted) vote for λ whenever it is in the partial label $\mathbf{y}_k^{\mathbf{t}}$.

Example 1. *Let us consider the case illustrated on Figure 1, where the training set contains* 6 *instances, the target set has* 5 *instances and output space* $\Omega = \{\lambda_1, \lambda_2, \lambda_3\}$.

Assuming that we work with $K = 3$, *the neighbours of each target instance and their associated (illustrative) weights are given in Table 1. And we have also:*

$$\mathbf{G}_{\mathbf{x}_1} = \mathbf{G}_{\mathbf{x}_2} = \mathbf{G}_{\mathbf{x}_3} = \{\mathbf{t}_1, \mathbf{t}_2\}$$
$$\mathbf{G}_{\mathbf{x}_4} = \mathbf{G}_{\mathbf{x}_5} = \mathbf{G}_{\mathbf{x}_6} = \{\mathbf{t}_3, \mathbf{t}_4, \mathbf{t}_5\}.$$

Precise instances $\mathbf{x}_5$ *and* $\mathbf{x}_6$ *cannot be queried, but which label among* $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ *and* $\mathbf{x}_4$ *should be queried is not obvious. Indeed,* $\mathbf{x}_4$ *is involved in more decisions than the three other partial labels (as* $|\mathbf{G}_{\mathbf{x}_4}|$ *is greater than all other sets), but getting more information about* $\mathbf{x}_4$ *will not change these decisions, as the result of Equation (1) will not change whatever the true label of* $\mathbf{x}_4$. *In contrast, knowing the true label of* $\mathbf{x}_1$, $\mathbf{x}_2$ *or* $\mathbf{x}_3$ *may change our decision about* $\mathbf{t}_1$ *and* $\mathbf{t}_2$, *hence from a decision viewpoint, querying these partial labels seems more interesting.*

In the next two sections, we will explore querying patterns following both intuitions (neighbour-based and ambiguity-based), but we first introduce a general querying scheme.

General Querying Scheme

Our general querying scheme follows a simple rule: for each partial label instance $\mathbf{x}_n$ and each target instance $\mathbf{t}$, we will define a function

$$f_{\mathbf{x}_n}(\mathbf{t}) \tag{2}$$

called local effect score, whose exact definition will vary for different criteria. The role of this function is to evaluate whether querying the training instance $\mathbf{x}_n$ can impact the result of the K-nn algorithm for the target instance $\mathbf{t}$. Since we want to improve the algorithm over the whole target set, this will be done by simply summing the effect of $\mathbf{x}_n$ over all data in the target set $\mathbf{T}$, that is by computing

$$f_{\mathbf{x}_n}(\mathbf{T}) = \sum_{\mathbf{t} \in \mathbf{T}} f_{\mathbf{x}_n}(\mathbf{t}), \tag{3}$$

that we will call global score function. The chosen instance to be queried, denoted by $\widehat{x_i}$, will then simply be the one with the highest effect score, or in other words

$$\widehat{x_i} = \arg\max_{\mathbf{x}_n \in \mathbf{D}} f_{\mathbf{x}_n}(\mathbf{T}).$$

We will now propose different ways to define $f_{\mathbf{x}_n}(\mathbf{t})$, that will be tested in the experimental section. Since the computation of the global effect score from the local ones is straightforward, we will focus in the next sections on computing $f_{\mathbf{x}_n}(\mathbf{t})$ for a single instance. Also, we will denote by q_n the query consisting in asking the true label of $\mathbf{x}_n$.

Neighbour-Based Querying Criteria

Our first idea is quite simple and consists in evaluating whether a partially labelled instance $\mathbf{x}_n$ is among the neighbours of the target instance $\mathbf{t}$, hence if it will participate to its classification, and how strongly it does so. This can be estimated by the simple function $f_{\mathbf{x}_n}^{MW}(\mathbf{t})$ as follows

$$f_{\mathbf{x}_n}^{MW}(\mathbf{t}) = \frac{w_n}{\sum_{k=1}^{K} w_k^{\mathbf{t}}} \tag{4}$$

where w_n is $w_k^{\mathbf{t}}$ if $\mathbf{x}_n$ is the kth neighbour of $\mathbf{t}$, and zero otherwise. The global effect score of $\mathbf{x}_n$ can then be computed using Equation (3). In the unweighted case, this score is the number of target instances of which $\mathbf{x}_n$ is a neighbour. This strategy is similar to the one of querying data in high-density regions in active learning techniques (Settles and Craven 2008).

Table 2 summarizes the global effect scores for Example 1. As expected, $\mathbf{x}_4$ is the one that should be queried according to f^{MW}, since it is the one participating to most decisions.

Table 2: Effect scores obtained by using f^{MW} in Example 1

$f^{MW}_{\mathbf{x}_n}$	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$	$\mathbf{t}_5$	$\mathbf{T}$
$\mathbf{x}_1$	0.4	0.2	0	0	0	0.6
$\mathbf{x}_2$	0.3	0.4	0	0	0	0.7
$\mathbf{x}_3$	0.3	0.4	0	0	0	0.7
$\mathbf{x}_4$	0	0	0.4	0.3	0.4	**1.1**

Indecision-Based Querying Criteria

This section presents other effect scores based on whether a partially labelled instance $\mathbf{x}_n$ introduces some ambiguity in the decision about an instance $\mathbf{t}$. We first define what we mean by ambiguity.

Ambiguous Instance: Definition

In the K-nn algorithm, each neighbour can be seen as a (weighted) voter in favor of his preferred class. Partial labels can then be assimilated to voters providing incomplete preferences. For this reason, we will define ambiguity by using ideas issued from plurality voting with incomplete preferences (Konczak and Lang 2005). More precisely, we will use the notions of necessary and possible winners of such a voting to determine when a decision is ambiguous.

For an instance $\mathbf{t}$ with a set of neighbours $\mathbf{N_t} = \{\mathbf{x}^{\mathbf{t}}_1, \dots, \mathbf{x}^{\mathbf{t}}_K\}$, we will denote by $\mathbf{L_t} = \{(l^{\mathbf{t}}_1, \dots, l^{\mathbf{t}}_K) | l^{\mathbf{t}}_k \in \mathbf{y}^{\mathbf{t}}_k\}$ the set of possible selections of $\mathbf{N_t}$ with cardinality $|\mathbf{L_t}| = \prod_{k=1}^{K} |\mathbf{y}^{\mathbf{t}}_k|$. For a given selection $\mathbf{l^t} \in \mathbf{L_t}$, the corresponding winner(s) of the voting procedure is (are)

$$\widehat{\lambda}_{\mathbf{l^t}} = \arg\max_{\lambda \in \Omega} \sum_{k=1}^{K} w^{\mathbf{t}}_k \mathbb{1}_{l^{\mathbf{t}}_k = \lambda}$$

with $w^{\mathbf{t}}_k$ the weight corresponding to the kth neighbor. Let us note that the $\arg\max$ can return multiple labels.

We can now define the possible ($\mathbf{PL_t}$) and necessary label sets ($\mathbf{NL_t}$) of $\mathbf{t}$ as follows:

$$\mathbf{PL_t} = \{\lambda \in \Omega | \exists \mathbf{l^t} \in \mathbf{L_t} \text{ s.t } \lambda \in \widehat{\lambda}_{\mathbf{l^t}}\} \tag{5}$$

and

$$\mathbf{NL_t} = \{\lambda \in \Omega | \forall \mathbf{l^t} \in \mathbf{L_t}, \lambda \in \widehat{\lambda}_{\mathbf{l^t}}\}, \tag{6}$$

which are nothing else but the set of possible and necessary winners in social choice theory. By definition, we have $\mathbf{NL_t} \subseteq \mathbf{PL_t}$. Given a target instance, we adopt the following definition of ambiguity.

Definition 1. *A target instance* $\mathbf{t}$ *is called ambiguous if* $\mathbf{NL_t} \neq \mathbf{PL_t}$.

By querying partial labels, we can reduce the ambiguity of $\mathbf{t}$ by either reducing $\mathbf{PL_t}$ or increasing $\mathbf{NL_t}$, eventually getting $\mathbf{PL_t} = \mathbf{NL_t}$ (see Proposition 3 further on).

Ambiguous Instance: Computation

A first issue is how to actually compute $\mathbf{NL_t}$ and $\mathbf{PL_t}$. The problem of determining $\mathbf{NL_t}$ is very easy (Konczak and Lang 2005). However, determining $\mathbf{PL_t}$ is in practice much more difficult. In the unweighted case, known results indicate (Betzler and Dorn 2010; Xia and Conitzer 2011) that $\mathbf{PL_t}$ can be determined in cubic (hence polynomial) time with respect to M, solving a maximum flow problem and using the fact that when votes are (made) unitary, the solution of this flow problem is integer-valued (due to the submodularity of the constraint matrix).

However, when votes are non-unitary (when weights are different), this result does not hold anymore, and the problem appears to be NP-complete, as it can be reduced to a 3-dimensional matching problem. A refined analysis of the complexity in terms of fixed parameters (M or K) could however help to identify those cases that are harder to solve from those that remain easy (polynomial). In addition to that, in our setting we can have to evaluate the set of possible labels $\mathbf{PL_t}$ a high number of times (in contrast with what happens in social choice, where $\mathbf{PL_t}$ and $\mathbf{NL_t}$ have to be evaluated at most a few times), hence even a cubic algorithm may have a prohibitive computational time. This is why we will provide an easy-to-compute approximation of it, denoted $\mathbf{APL_t}$. Let us first provide some definitions.

Table 3: Minimal and maximal scores for Example 1

λ	Scores	$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$	$\mathbf{t}_4$	$\mathbf{t}_5$
λ_1	$S^{\min}$	0	0	1.2	1.4	1.2
	$S^{\max}$	0.7	0.8	2	2.1	2
λ_2	$S^{\min}$	0	0	0	0	0
	$S^{\max}$	1.7	1.2	0.8	0.7	0.8
λ_3	$S^{\min}$	0	0	0	0	0
	$S^{\max}$	2.4	2	0	0	0

Given the set of nearest neighbours $\mathbf{N_t}$, we denote by $\Omega_{\mathbf{t}} = \cup_{k=1}^{K} \mathbf{y}^{\mathbf{t}}_k \subseteq \Omega$ all labels included in the neighbours of $\mathbf{t}$. For each label $\lambda \in \Omega_{\mathbf{t}}$, we define the minimum and maximum scores as

$$S^{\min}(\lambda) = \sum_{k=1}^{K} w^{\mathbf{t}}_k \mathbb{1}_{\lambda = \mathbf{y}^{\mathbf{t}}_k} \text{ and } S^{\max}(\lambda) = \sum_{k=1}^{K} w^{\mathbf{t}}_k \mathbb{1}_{\lambda \in \mathbf{y}^{\mathbf{t}}_k},$$

respectively. For a given selection $\mathbf{l}^t$, we also denote by $S^{\mathbf{l^t}}(\lambda) = \sum_{k=1}^{K} w^{\mathbf{t}}_k \mathbb{1}_{\lambda = l^{\mathbf{t}}_k}$ the score received by λ. For any $\mathbf{l^t}$, we can see that

$$S^{\min}(\lambda) \leq S^{\mathbf{l^t}}(\lambda) \leq S^{\max}(\lambda). \tag{7}$$

$S^{\min}(\lambda)$ and $S^{\max}(\lambda)$ are therefore the minimal and maximal scores that the candidate λ can receive.

Table 3 provides the score bounds obtained for the different $\mathbf{t}_i$ of Example 1.

From the minimal and maximal scores, we can easily get $\mathbf{NL_t}$ and an approximation ($\mathbf{APL_t}$) of $\mathbf{PL_t}$, as indicated in the next proposition and definition. All proofs are given in the supplementary material.

Proposition 1. *Given target instance* $\mathbf{t}$*, weight* $\mathbf{w_t}$ *and nearest neighbour set* $\mathbf{N_t}$*, a label* $\lambda \in \mathbf{NL_t}$ *iff*

$$S^{min}(\lambda) \geq S^{max}(\lambda^{'}), \forall \lambda^{'} \neq \lambda, \lambda^{'} \in \Omega_{\mathbf{t}}. \tag{8}$$

Definition 2. *Given target instance* $\mathbf{t}$*, weight* $\mathbf{w_t}$ *and nearest neighbour set* $\mathbf{N_t}$*, a label* $\lambda \in \mathbf{APL_t}$ *iff*

$$S^{max}(\lambda) \geq \max_{\lambda' \in \Omega_\mathbf{t}} S^{min}(\lambda'), \forall \lambda' \neq \lambda, \lambda' \in \Omega_\mathbf{t}. \quad (9)$$

Example 2. *According to Table 3, the sets obtained for Example 1 with* $K = 3$ *are*

$$\begin{aligned}\mathbf{NL_{t}}_3 = \mathbf{NL_{t}}_4 = \mathbf{NL_{t}}_5 = \mathbf{APL_{t}}_3 \\ = \mathbf{APL_{t}}_4 = \mathbf{APL_{t}}_5 = \{\lambda_1\}\end{aligned}$$

and

$$\mathbf{NL_{t}}_1 = \mathbf{NL_{t}}_2 = \emptyset \qquad \mathbf{APL_{t}}_1 = \mathbf{APL_{t}}_2 = \{\lambda_1, \lambda_2, \lambda_3\},$$

showing, as expected, that only $\mathbf{t}_1, \mathbf{t}_2$ *are ambiguous.*

The next proposition states that $\mathbf{APL_t}$ is an outer approximation of $\mathbf{PL_t}$ (therefore not missing any possible answer) and that both coincide whenever $\mathbf{NL_t}$ is non-empty (therefore guaranteeing that using $\mathbf{APL_t}$ will not make some instance artificially ambiguous).

Proposition 2. *Given target instance* $\mathbf{t}$*, weight* $\mathbf{w_t}$ *and nearest neighbour set* $\mathbf{N_t}$*, the following properties hold*

A1. $\mathbf{APL_t} \supseteq \mathbf{PL_t} \supseteq \mathbf{NL_t}$

A2. *if* $\mathbf{NL_t} \neq \emptyset$*, then* $\mathbf{APL_t} = \mathbf{PL_t}$.

Effect of a Query on Ambiguous Instances

Now that we have defined how to identify an ambiguous instance, the question arises as to how we can identify queries that will help to reduce this ambiguity. This section provides some answers by using the notions of necessary and (approximated) possible labels to define a local effect score. More precisely, the local effect score $f_{\mathbf{x}_n}(\mathbf{t})$ will take value one if a query can modify either the sets $\mathbf{PL_t}$ or $\mathbf{APL_t}$, or the set $\mathbf{NL_t}$. Additionally, as this local effect score aims at detecting whether a query can affect the final decision, it will also take value one if it can change the decision $h(\mathbf{t})$ taken by Equation (1). In some sense, such a strategy is close to active learning techniques aiming to identify the instances for which the decision is the most uncertain (uncertainty sampling (Lewis and Catlett 1994), query-by-committee (Seung, Opper, and Sompolinsky 1992)).

To define this score, we need to know when a query q_n can potentially change the values of the possible label set $\mathbf{PL_t}$, the approximated possible label set $\mathbf{APL_t}$, the prediction set $h(\mathbf{t})$ or the necessary label set $\mathbf{NL_t}$. A first remark is that if an instance $\mathbf{x}_n \notin \mathbf{N_t}$ is not among the neighbours of $\mathbf{t}$, then a query q_n cannot change any of these values. Let us now investigate the conditions under which q_n can change the sets when $\mathbf{x}_n \in \mathbf{N_t}$. We first introduce some useful relations between the sets $\mathbf{PL_t}$, $\mathbf{APL_t}$, or $\mathbf{NL_t}$. We will denote by $\mathbf{PL}_\mathbf{t}^{q_n}$, $\mathbf{APL}_\mathbf{t}^{q_n}$, and $\mathbf{NL}_\mathbf{t}^{q_n}$ the sets potentially obtained once $\mathbf{x}_n$ is queried.

Proposition 3. *Given target instance* $\mathbf{t}$*, set of nearest neighbours* $\mathbf{N_t}$ *and query* q_n*, then the following properties hold*

B1. $\mathbf{APL}_\mathbf{t}^{q_n} \subseteq \mathbf{APL_t}$

B2. $\mathbf{NL_t} \subseteq \mathbf{NL}_\mathbf{t}^{q_n} \subseteq \mathbf{PL}_\mathbf{t}^{q_n} \subseteq \mathbf{PL_t}$

B3. *Furthermore, if* $\mathbf{NL_t} = \mathbf{PL_t} = \mathbf{APL_t}$*, then*

$$\mathbf{NL}_\mathbf{t}^{q_n} = \mathbf{PL}_\mathbf{t}^{q_n} = \mathbf{APL}_\mathbf{t}^{q_n} = \mathbf{NL_t},$$

meaning that $\mathbf{APL_t}$*,* $\mathbf{PL_t}$*,* $\mathbf{NL_t}$ *cannot change after* q_n.

In this paper, a query will be considered interesting (i.e., having a local effect score of one) if at least one value $\lambda \in \mathbf{y}_n$ can change $\mathbf{NL_t}$, $\mathbf{PL_t}$, $\mathbf{APL_t}$ or $h(\mathbf{t})$. Indeed, requiring all possible values $\lambda \in \mathbf{y}_n$ to change the sets of necessary labels $\mathbf{NL_t}$, possible labels $\mathbf{PL_t}$, approximation $\mathbf{APL_t}$ or prediction set $h(\mathbf{t})$ is much too demanding, and is unlikely to happen in practice.

We will go from the cases that are the most likely to happen in practice, that is changes in $\mathbf{PL_t}$ or $\mathbf{APL_t}$, to the most unlikely cases, that is changes in $\mathbf{NL_t}$. The next proposition investigates conditions under which $\mathbf{APL_t}$ will not change.

Proposition 4. *Given target instance* $\mathbf{t}$*, nearest neighbour set* $\mathbf{N_t}$*, set* $\mathbf{APL_t}$ *of labels of the neighbours of* $\mathbf{t}$ *and weight* $\mathbf{w_t}$*, query* q_n *cannot change* $\mathbf{APL_t}$ *if the two following conditions hold*

C1. *for any* $\lambda \in \mathbf{APL_t}$ *and* $\lambda \notin \mathbf{y}_n$

$$S^{max}(\lambda) \geq \max_{\lambda' \in \mathbf{y}_n} S^{min}(\lambda') + w_n.$$

C2. *and for any* $\lambda \in \mathbf{APL_t} \cap \mathbf{y}_n$*, we have*

$$S^{max}(\lambda) - w_n \geq$$
$$\max\big(\max_{\lambda' \in \mathbf{y}_n \setminus \{\lambda\}} S^{min}(\lambda') + w_n, \max_{\lambda' \in \Omega_\mathbf{t} \setminus \mathbf{y}_n} (S^{min}(\lambda'))\big).$$

According to Equation (9), a label $\lambda \notin \mathbf{APL_t}$ if there is a label λ' whose minimal score $S^{\min}(\lambda')$ is higher than $S^{\max}(\lambda)$. Proposition 4 identifies, for a label $\lambda \in \mathbf{APL_t}$, those conditions under which an increase of the minimal score $S^{\min}(\lambda')$ for other labels is not sufficient to become higher than $S^{\max}(\lambda)$. Otherwise, λ could get out of $\mathbf{APL_t}$.

The case of $\mathbf{PL_t}$ is more complex, and since estimating it requires to enumerate selections, the same goes for evaluating whether a query can change it. In particular, we could not find any simple-to-evaluate conditions (as those of Proposition 4) to check whether a query can change $\mathbf{PL_t}$, and we are reduced to provide the following definition. This means that evaluating whether a query can change the set $\mathbf{PL_t}$ will only be doable when K or the cardinality of partial labels neighbours will be small.

Definition 3. *Given partial label* $\mathbf{y}_n$*, nearest neighbours* $\mathbf{N_t}$*, possible label set* $\mathbf{PL_t}$*, set* $\Omega_\mathbf{t}$ *and weight* $\mathbf{w_t}$*, a query* q_n *on* $\mathbf{x}_n \in \mathbf{N_t}$ *is said to not affect* $\mathbf{PL_t}$ *if, for every possible answer* $\lambda \in \mathbf{y}_n$ *of the query, we have* $\mathbf{PL}_\mathbf{t}^{q_n=\lambda} = \mathbf{PL_t}$*, where* $\mathbf{PL}_\mathbf{t}^{q_n=\lambda}$ *denotes the set* $\mathbf{PL}^{q_n}$ *when* $\mathbf{y}_n = \lambda$.

The next proposition investigates whether or not a query can change the decision given by Equation (1) that we use to make predictions from partially labelled neighbours.

Proposition 5. *Given target instance* $\mathbf{t}$*, nearest neighbour set* $\mathbf{N_t}$*, prediction set* $h(\mathbf{t})$*, label set* $\Omega_\mathbf{t}$ *and weight* $\mathbf{w_t}$*, query* q_n *does not affect* $h(\mathbf{t})$ *if at least one of following conditions hold*

D1. $h(\mathbf{t}) \cap \mathbf{y}_n = \emptyset$.

D2. $\forall \lambda \in h(\mathbf{t}) \cap \mathbf{y}_n$,

$$S^{max}(\lambda) - w_n > \max_{\lambda' \in \Omega_{\mathbf{t}} \setminus \{\lambda\}} S^{max}(\lambda') \tag{10}$$

Since classifier h takes decisions based on the maximal number of votes a label can receive, this proposition simply identifies the cases where the reduced score of $S^{\max}(\lambda)$ with $\lambda \in h(\mathbf{t})$ (or non-reduction in case D1) cannot become smaller than another $S^{\max}(\lambda')$. Finally, we give some conditions under which $\mathbf{NL_t}$ will not change, which may happen in practice.

Proposition 6. *Given target instance* $\mathbf{t}$*, nearest neighbour set* $\mathbf{N_t}$*, necessary label set* $\mathbf{NL_t}$ *and weight* $\mathbf{w_t}$*, then query* q_n *cannot change* $\mathbf{NL_t}$ *if the two following conditions hold*

E1. *for any* $\lambda \notin \mathbf{NL_t}$ *and* $\lambda \notin \mathbf{y}_n$*,*

$$S^{min}(\lambda) < \max\Big(\max_{\lambda' \neq \lambda, \lambda' \in \Omega_{\mathbf{t}} \setminus \mathbf{y}_n} S^{max}(\lambda'), \max_{\lambda' \in \mathbf{y}_n} S^{max}(\lambda') - w_n, \min_{\lambda' \in \mathbf{y}_n} S^{max}(\lambda') \Big),$$

E2. *for any* $\lambda \notin \mathbf{NL_t}$ *and* $\lambda \in \mathbf{y}_n$

$$S^{min}(\lambda) + w_n < \max\Big(\max_{\lambda' \notin \mathbf{y}_n} S^{max}(\lambda'), \max_{\lambda' \in \mathbf{y}_n \setminus \{\lambda\}} S^{max}(\lambda') - w_n \Big).$$

According to Equation (8), a label $\lambda \in \mathbf{NL_t}$ if its minimal score $S^{\min}(\lambda)$ is higher than the maximal scores of all the other labels λ'. Proposition 6 identifies, for a given label $\lambda \in \Omega_{\mathbf{t}}$, the conditions under which a decrease of the maximal score $S^{\max}(\lambda')$ of the other labels is not sufficient to become lower than $S^{\min}(\lambda)$ (otherwise, λ could be included in $\mathbf{NL_t}$ after the query). Condition E1 covers the cases where λ is certainly not the true label, while condition E2 covers the cases where it may be the true label.

We can now use those propositions and definition to define the two local effect scores measuring whether querying $\mathbf{x}_n$ can impact our decision on $\mathbf{t}$:

$$f^{PL}_{\mathbf{x}_n}(\mathbf{t}) = \begin{cases} 0 \text{ if Def. 3, Prop. 5, Prop. 6 hold} \\ \frac{w_n}{\sum_{k=1}^{K} w_k^{\mathbf{t}}} \text{ otherwise.} \end{cases} \tag{11}$$

and

$$f^{APL}_{\mathbf{x}_n}(\mathbf{t}) = \begin{cases} 0 \text{ if Prop. 4, Prop. 5, Prop. 6 hold} \\ \frac{w_n}{\sum_{k=1}^{K} w_k^{\mathbf{t}}} \text{ otherwise.} \end{cases} \tag{12}$$

In the next sections, query schemes corresponding to $f^{PL}_{\mathbf{x}_n}(\mathbf{t})$ and $f^{APL}_{\mathbf{x}_n}(\mathbf{t})$ are denoted shortly by PL and APL, respectively. Since $f^{PL}_{\mathbf{x}_n}(\mathbf{t})$ uses exact information to identify the ambiguous instances, we can expect the model accuracy to improve faster by using it, yet getting $f^{PL}_{\mathbf{x}_n}(\mathbf{t})$ is computationally demanding. In practice, $f^{APL}_{\mathbf{x}_n}(\mathbf{t})$ offers a cheap approximation that can still provide good results (this will be confirmed by our experiments).

Tables 4 and 5 provide an overview of the computations associated to Example 1. Each time a proposition does not hold, we provide between parenthesis the specific answer for which it does not hold.

Table 4: Check for propositions for Example 1.

		$\mathbf{APL_t}$	$h(\mathbf{t})$	$\mathbf{NL_t}$	$\mathbf{PL_t}$
		Prop. 4	Prop. 5	Prop. 6	Def. 3
$\mathbf{t}_1$	$\mathbf{x}_1$	No (λ_3)	No (λ_2)	No (λ_3)	No (λ_3)
	$\mathbf{x}_2$	No (λ_3)	Yes	Yes	Yes
	$\mathbf{x}_3$	No (λ_2)	No (λ_2)	Yes	Yes
$\mathbf{t}_2$	$\mathbf{x}_1$	Yes	No (λ_2)	Yes	Yes
	$\mathbf{x}_2$	Yes	No (λ_1)	Yes	No (λ_3)
	$\mathbf{x}_3$	No (λ_3)	No (λ_3)	No (λ_3)	No (λ_3)

Table 5: Ambiguity effect for Example 1.

	$f^{PL}_{\mathbf{x}_n}(\mathbf{t}_1)$	$f^{APL}_{\mathbf{x}_n}(\mathbf{t}_1)$	$f^{PL}_{\mathbf{x}_n}(\mathbf{t}_2)$	$f^{APL}_{\mathbf{x}_n}(\mathbf{t}_2)$
$\mathbf{x}_1$	0.4	0.4	0.2	0.2
$\mathbf{x}_2$	0	0.3	0.4	0.4
$\mathbf{x}_3$	0.3	0.3	0.4	0.4

From Table 5, we can see that $f^{APL}_{\mathbf{x}_2}(\mathbf{T}) = f^{APL}_{\mathbf{x}_3}(\mathbf{T}) = 0.7$, but that $f^{PL}_{\mathbf{x}_2}(\mathbf{T}) = 0.4$ and $f^{PL}_{\mathbf{x}_3}(\mathbf{T}) = 0.7$, meaning that the two effect scores given by Equations (12) and (11) would provide different results. Finally, note that since $f^{PL}_{\mathbf{x}_n}(\mathbf{t})$ and $f^{APL}_{\mathbf{x}_n}(\mathbf{t})$ will be positive as soon as only one proposition or definition does not hold, we do not need to evaluate all of them if we know that one does not hold.

Experiments

This section presents the experimental setup and the results obtained with benchmark data sets which are used to illustrate the behaviour of the proposed schemes.

Experimental Setup

We do experiments on "contaminated" versions of standard, precise benchmark data sets. To contaminate a given data set, we used two methods (Hüllermeier and Beringer 2006):

Random Model: Each training instance is contaminated randomly with probability p. In case an example $\mathbf{x}_n$ is contaminated, the set $\mathbf{y}_n$ of candidate labels is initialized with the original label λ, and all other labels $\lambda' \in \Omega \setminus \{\lambda\}$ are added with probability q, independently of each other.

Bayes Model: In order to take the dependencies between labels (more likely to happen in practice) into account, a second approach is used. First, a Naive Bayes classifier is trained using the original data (precise labels) so that each label is associated to a posterior probability $Pr(\lambda|\mathbf{x}_n)$. As before, each training instance will be contaminated randomly at probability p. In case of contamination, the true label is retained, the other labels are re-arranged according to their probabilities and the k^{th} label is included in the set of labels with probability $\frac{2kq}{|\Omega|}$.

Note that in Bayes model, the probability $^{2kq}/_{|\Omega|}$ can exceed 1 when parameter q is greater than 0.5. However, this value of $^{2kq}/_{|\Omega|}$ ensures that the expected cardinality of the partial labels, in case of contamination, is $1 + (M-1)q$ for both contamination models, making them comparable

Table 6: Data set used in the experiments

Name	# instances	# features	# labels
iris	150	4	3
wine	178	13	3
forest	198	27	4
seeds	210	7	3
glass	214	9	6
ecoli	336	7	8
libras	360	91	15
dermatology	385	34	6
vehicle	846	18	4
vowel	990	10	11
yeast	1484	8	12
winequality	1599	11	6
optdigits	1797	64	10
segment	2300	19	7
wall-following	5456	24	4

Table 7: Complexities of query schemes

RD	MP/ACT	MW	APL	PL
$\mathcal{O}(1)$	$\mathcal{O}(T)$	$\mathcal{O}(TK)$	$\mathcal{O}(TM(M+K))$	$\mathcal{O}(TM^K)$

Table 8: Average error rates % (average ranks) over the 15 data sets

K	Scheme	Random	Bayes	Random	Bayes
		$p = 0.7$	$p = 0.7$	$p = 0.9$	$p = 0.9$
		$q = 0.5$	$q = 0.5$	$q = 0.9$	$q = 0.9$
3	no query	36.4	42.6	77.8	78.8
	RD	30.8(4.60)	34.6(4.40)	61.6(3.73)	62.4(3.33)
	MP	29.9(3.60)	34.3(4.20)	62.4(4.13)	63.1(3.93)
	ACT	32.6(5.53)	37.5(5.73)	66.2(5.33)	66.5(5.07)
	MW	27.6(2.33)	29.9(2.73)	54.0(2.20)	54.2(1.53)
	APL	27.3(1.67)	29.4(1.67)	53.5(1.53)	54.1(1.60)
	PL	27.2(1.27)	29.3(1.33)	53.5(1.33)	54.1(1.60)
6	no query	25.7	30.4	63.3	65.6
	RD	24.0(3.40)	26.4(3.53)	44.9(3.27)	45.6(3.27)
	MP	23.7(2.00)	26.0(3.00)	45.6(3.80)	46.6(3.60)
	ACT	24.4(3.87)	27.8(4.87)	51.2(4.93)	52.7(4.93)
	MW	23.6(2.40)	25.0(2.07)	37.8(1.87)	38.9(1.73)
	APL	23.4(1.53)	24.6(1.07)	36.0(1.13)	37.5(1.20)
9	no query	25.4	27.9	53.7	57.5
	RD	24.4(2.47)	25.5(2.67)	37.0(2.93)	38.2(2.80)
	MP	24.1(1.53)	25.6(2.93)	38.3(3.73)	39.8(3.67)
	ACT	24.6(3.07)	26.5(4.40)	43.4(4.73)	45.8(4.87)
	MW	24.5(3.07)	25.8(2.47)	33.7(2.33)	34.8(2.33)
	APL	24.3(2.40)	25.6(1.73)	31.7(1.13)	33.3(1.33)

(Hüllermeier and Beringer 2006). In practice, we lowered $^{2kq}/_{|\Omega|}$ to 1 once it go over it.

Results have been obtained for 15 UCI data sets described in Table 6. Three different values for K (3, 6 and 9) have been used for all experiments. The weight $w_k^{\mathbf{t}}$ for an instance $\mathbf{t}$ is $w_k^{\mathbf{t}} = 1 - (d_k^{\mathbf{t}})/(\sum_{j=1}^{K} d_j^{\mathbf{t}})$ with $d_j^{\mathbf{t}}$ the Euclidean distance between $\mathbf{x}_j^{\mathbf{t}}$ and $\mathbf{t}$. As usual when working with Euclidean distance based K-nn, data is normalized.

We use a three-fold cross-validation procedure: each data set is randomly split into 3 folds. Each fold is in turn considered as the test set, the other folds are used for the training set. The training set is contaminated according to one of the models with two combinations of (p, q) parameters:($p = 0.7$, $q = 0.5$) and ($p = 0.9$, $q = 0.9$), which correspond to low and high levels of partiality. The error rate is computed as the average error obtained from the 3 test sets. This process is repeated 10 times and results are also averaged. For each data set, the number of queries I has been fixed to 10% of the number of training data.

Similarly to what is done in active learning, we pick $\mathbf{T}$ (the pool of unlabelled data) as the set of partially labelled instances.

To evaluate the efficiency of the proposed query schemes (MW, PL and APL), we compare our results with 3 baseline schemes:

- RD: a query is picked up at random from the pool;
- MP: the one with the largest partial label is picked up;
- ACT: partially labelled instances are considered as unlabeled ones and ModFF, a classical active learning scheme (Joshi, Porikli, and Papanikolopoulos 2012), is used to query instances. ModFF selects the queries in such a way that all target data have labelled samples at a bounded maximum distance.

The complexity of each scheme for a single query is given in Table 7. Note that the more computationally demanding PL scheme was only tested for the case $K = 3$.

Results

For each scheme, the error rate after querying 10% of the number of training data has been computed and the schemes have been ranked according to this error rate. The average error rates and the average ranks of the schemes over the 15 data sets are given in Table 8. Complete results, as well as detailed results of all statistical tests are given in supplementary material.

A Friedman test done over the ranks indicates that, in all settings, there are significant evidence that not all algorithms are equivalent (except for the random setting with low partiality that gave a p-value of 0.002, all other are below 10^{-5}). Nemenyi post-hoc test performed to identify the differences between the schemes indicate that our proposed schemes (MW, PL, APL) work almost systematically better than any baseline, with APL having a significant positive difference in pairwise tests. A noticeable exception is when the partiality is low and $K = 9$. However in this case it can be seen from Table 8 that all querying techniques only improve results in a very marginal way (with an accuracy gain around 1% for all methods).

A second look at Table 8 confirms that the proposed methods really provide an edge (in terms of average accuracy gain) in the situations where ambiguous situations are the most present, that is when:

- K is low, in which case even a few partial labels among the neighbours may lead to ambiguous situations, a fact that is much less likely when K gets higher.

- There is a large amount of partial labels, in which case increasing the value of K will have a very limited effect on the number of ambiguous cases.

Both cases are of practical interest, as even if picking a higher value of K is desirable when having low partiality, it may be computationally unaffordable.

Finally, we can notice that the Bayes contamination induces slightly more ambiguity in the data sets, as more likely classes (hence similar labels in a given region of the input space) have more chances to appear in the contaminated labels. Bayes contamination also seem somehow more realistic, as experts or labellers will have a tendency to provide sets of likely labels as partial information.

Conclusion

This paper has adressed the problem of selecting the instances that should be disambiguated in order to improve the accuracy of a K-nn classifier, when several training instances are characterized by imprecise labels. We have proposed two querying schemes, both based on the computation of an effect score quantifying the impact of a disambiguation on the final result. A first strategy (neighbour-based) consists in selecting an instance when it is involved in many decisions. A more refined strategy (indecision-based) consists in selecting an instance when it can potentially reduce the ambiguity of one or several decisions. This second strategy is more complex from a computational point of view, and we have therefore proposed an approximate scheme leading to very close performance. The experiments have shown that the accuracy of K-nn classifiers is significantly improved by querying partial label instances and that indecision-based querying strategies are the best-performing schemes.

Acknowledgement

This work was carried out in the framework of Labex MS2T, which was funded by the French National Agency for Research (Reference ANR-11-IDEX-0004-02).

References

Betzler, N., and Dorn, B. 2010. Towards a dichotomy for the possible winner problem in elections based on scoring rules. *Journal of Computer and System Sciences* 76(8):812–836.

Cour, T.; Sapp, B.; Jordan, C.; and Taskar, B. 2009. Learning from ambiguously labeled images. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 919–926. IEEE.

Cour, T.; Sapp, B.; and Taskar, B. 2011. Learning from partial labels. *The Journal of Machine Learning Research* 12:1501–1536.

Cover, T., and Hart, P. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory* 13(1):21–27.

Hüllermeier, E., and Beringer, J. 2006. Learning from ambiguously labeled examples. *Intelligent Data Analysis* 10(5):419–439.

Joshi, A. J.; Porikli, F.; and Papanikolopoulos, N. 2012. Coverage optimized active learning for k-nn classifiers. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 5353–5358. IEEE.

Konczak, K., and Lang, J. 2005. Voting procedures with incomplete preferences. In *Proc. IJCAI-05 Multidisciplinary Workshop on Advances in Preference Handling*, volume 20. Citeseer.

Lewis, D. D., and Catlett, J. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the eleventh international conference on machine learning*, 148–156.

Liu, L., and Dietterich, T. 2014. Learnability of the superset label learning problem. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 1629–1637.

Moulin, H.; Brandt, F.; Conitzer, V.; Endriss, U.; Lang, J.; and Procaccia, A. D. 2016. *Handbook of Computational Social Choice*. Cambridge University Press.

Settles, B., and Craven, M. 2008. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing*, 1070–1079. Association for Computational Linguistics.

Seung, H. S.; Opper, M.; and Sompolinsky, H. 1992. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, 287–294. ACM.

Wu, X.; Kumar, V.; Quinlan, J. R.; Ghosh, J.; Yang, Q.; Motoda, H.; McLachlan, G. J.; Ng, A.; Liu, B.; Philip, S. Y.; et al. 2008. Top 10 algorithms in data mining. *Knowledge and information systems* 14(1):1–37.

Xia, L., and Conitzer, V. 2011. Determining possible and necessary winners given partial orders. *Journal of Artificial Intelligence Research* 25–67.

Yang, F., and Vozila, P. 2014. Semi-supervised chinese word segmentation using partial-label learning with conditional random fields. In *EMNLP*, 90–98.

Zhang, M.-L., and Yu, F. 2015. Solving the partial label learning problem: an instance-based approach. In *Proceedings of the 24th International Conference on Artificial Intelligence*, 4048–4054. AAAI Press.