This is an author manuscript of the paper:

Georg Krempl, Daniel Kottke, Myra Spiliopoulou.
Probabilistic Active Learning: Towards Combining Versatility, Optimality & Efficiency.
In: S. Deroski, P. Panov, D. Kocev, and L. Todorovski (eds.). Proceedings of the 17th International Conference on Discovery Science (DS), October 8–10, 2014, Bled, Slovenia. Lecture Notes in Computer Science, Springer. ISSN: 0302-9743.
The original publication is available at http://link.springer.com

For a companion website to this paper, please consult:
http://kmd.cs.ovgu.de/res/pal/

Probabilistic Active Learning: Towards Combining Versatility, Optimality and Efficiency

Georg Krempl[1], Daniel Kottke[1], and Myra Spiliopoulou[1]

Knowledge Management and Discovery Lab, University Magdeburg, Germany,
[georg.krempl|daniel.kottke|myra]@iti.cs.uni-magdeburg.de,
WWW home page: kmd.cs.ovgu.de

Abstract. Mining data with minimal annotation costs requires efficient active approaches, that ideally select the optimal candidate for labelling under a user-specified classification performance measure. Common generic approaches, that are usable with any classifier and any performance measure, are either slow like error reduction, or heuristics like uncertainty sampling. In contrast, our Probabilistic Active Learning (PAL) approach offers versatility, direct optimisation of a performance measure and computational efficiency. Given a labelling candidate from a pool, PAL models both the candidate's label and the true posterior in its neighbourhood as random variables. By computing the expectation of the gain in classification performance over both random variables, PAL then selects the candidate that in expectation will improve the classification performance the most. Extending our recent poster, we discuss the properties of PAL and perform a thorough experimental evaluation on several synthetic and real-world data sets of different sizes. Results show comparable or better classification performance than error reduction and uncertainty sampling, yet PAL has the same asymptotic time complexity as uncertainty sampling and is faster than error reduction.

1 Introduction

Recently, the application of machine learning to large data pools and fast data streams has gained attention. This application often requires classification of data where features are cheap but labels are costly [8]. Examples are applications where features are obtained from an automated process but labels require human annotation efforts. Active learning (AL) [15, p. 4] addresses such applications, where the machine learning system can actively select instances for labelling, rather than passively processing a given set of labelled instances. Its tasks are to decide a) for which instance to request a label, and b) whether to continue labelling at all, given some labels have already been acquired.

The ideal active learning strategy should select those instances first that, once incorporated into the training data, will result in the highest gain in terms of a classification performance measure. Furthermore, it provides a quantification of this performance gain, needed for a sound answer to the stop-criterion related second question. It therefore considers the already acquired amount of training data. Finally, it is fast, requiring solely linear asymptotic computational time per instance with respect to the pool size, in order to enable its application in large data pools and fast data streams. Active learning strategies that are usable in conjunction with any classifier technology provide some

of the above qualities. However, as discussed further in Section 2, they do not offer a *combination of all these qualities* in one single approach.

We address this challenge by a novel, probabilistic active learning (PAL) technique for classification that combines the above qualities and constitutes an alternative to other generic strategies like error reduction or uncertainty sampling. It is not limited to a particular classifier technology, and usable with any point [12] performance performance measure. Given a pool of candidates, it computes for each candidate the expected gain in classification performance from obtaining its label. This expectation models the candidate's label *and* the true posterior at its location as a random variables, and uses likelihood weights according to the already obtained labels in the candidate's neighbourhood. Subsequently, it selects the optimal candidate under this expected overall performance gain for labelling. This active selection from a pool requires asymptotic computational time that is solely linear in the size of the pool, as fast uncertainty sampling approaches do. While deriving stop-criteria is not within the scope of this paper, but our quantification of a label's expected impact provides a fundamental first step.

This paper is a full-version of our recent poster [10], extending it by a more detailed discussion of related work, an additional discussion of PAL's properties, and additional experiments. It is structured as follows: In the next section, we provide the necessary background and discuss related approaches. In section 3, we present our probabilistic active learning approach. In section 4, we report on our evaluation results, where we compare *PAL* to the strategy considered to be optimal for minimising classification error (error reduction), and to a popular fast heuristic strategy (uncertainty sampling). [1]

2 Background and Related Work

This paper addresses *pool-based* active learning (AL) for binary classifiers, as described in [15, p. 9] and [4]. In this scenario, an active classifier has access to a pool of unlabelled instances $\mathcal{U} = \{(x, .)\}$. From this pool of labelling candidates it repeatedly selects an instance $(x^*, .)$ for labelling. Upon receiving its label y^*, the instance (x^*, y^*) is moved to a pool of labelled instances $\mathcal{L} = \{(x, y)\}$, the classifier is retrained, and the process is repeated. There exist various approaches for this scenario, recent surveys are provided in [15], [6], [4] and [14]. We will focus on popular families of approaches that are usable with any classification technique, and discuss the ones most related to our approach: error reduction, uncertainty sampling and query-by-transduction.

Expected error reduction (ER) is a decision-theoretic approach. It considers the improvement in classification performance by selecting the candidate, that has the minimal expected classification error if incorporated into the training pool. The seminal work of [5], which coined the term "statistically optimal active learning", derived closed-form solutions for optimal data selection for two specific learning methods. In contrast, the approach suggested in [13] is generic, both with respect to arbitrary performance measures and classifiers: using a Monte Carlo sampling approach, it estimates the performance on a labelled validation sample $\mathcal{V}$, rather than integrating over the full feature distribution $\mathcal{P}r(x)$. It uses the posterior estimate $\hat{p} = \hat{\mathcal{P}}r(y|x)$ provided by the current

[1] For additional resources please consult `http://kmd.cs.ovgu.de/res/pal/`.

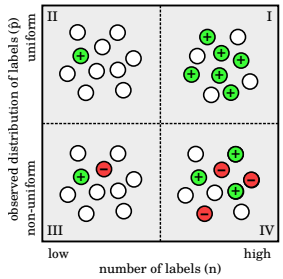
classifier as proxy for the true posterior $\mathcal{P}r(y|x)$ that is required for the expectation over the label realisations y. However, as discussed in [2], this proxy is not reliable if solely few labels are available, requiring regularisation approaches such as using Beta priors. Furthermore, the labelled (or self-labelled) validation sample $\mathcal{V}$ must be representative of the data. Not only is this difficult, in particular at the beginning with few available labels and a still unreliable classifier, but it also makes error reduction prohibitively slow [14] for using it in applications that require fast processing of big amounts of data, as even for incremental classifiers its asymptotic time complexity is $O(|\mathcal{V}| \cdot |\mathcal{U}|)$.

In comparison, a faster method [15, p.64] is uncertainty sampling (US), introduced in [11]. It uses simple uncertainty measures, like sample margin, confidence, or entropy as proxies for a candidate's value, and selects the candidate with maximal uncertainty. However, these proxies do not consider the number of similar instances on which posterior estimates are made. This is problematic, as Figure 1 (next page) illustrates on four exemplary active learning situations. These situations could, for example, occur simultaneously in different regions of a feature space such that the next label must be actively requested in either of them[2] The first (in Roman numeral) and second situation differ in the number of obtained labels (6 vs. 1), but lead to the same posterior estimate $\hat{\mathcal{P}}r(+|x) = 1$, as all obtained labels are positive. Uncertainty sampling is indifferent between them, as both entropy and confidence are zero. This indicates equal and absolute certainty, which is not justified as in II the single positive label can simply be due to chance, even if the true posterior of the positive class is actually smaller than 0.5 and the classifier is wrong. In contrast, in I a high true positive posterior is indeed very likely, and additional labels have less impact on the classifier. Similarly, in IV the classifier's prediction is quite reliable, but uncertainty according to measures like entropy or confidence is maximal. This leads to sampling in regions of high Bayesian error rate, even if the classifier can not be further improved there.

Some of the many existing *classifier-specific* AL approaches offer high processing speeds for particular applications. However, they require classifier selection to be made with respect to the available active learning strategy, as sample reusability between different types of classifiers for selector and consumer strategies is an open question [16]. Finally, even recently proposed classifier-specific approaches are mostly either information-theoretic (i.e. agnostic to the decision task at hand) or use the most likely or most pessimistic posterior under the current model, thus ignoring the reliability associated with this estimate, as for example [7]. A very recent information-theoretic approach that considers the reliability of a predictive model is Query-By-Transduction (QbT) [9]. QbT is based on conformal prediction and selects the instances with respect to the p-values obtained using transduction. This quantification of the reliability using p-values is related to ours, although we use the likelihood weights of the posterior estimates and follow a decision-theoretic Bayes-optimal active learning approach that directly optimises a classification performance measure.

[2] For simplicity, this illustration assumes conditional independence of the posterior from the feature given the region, i.e. $\mathcal{P}r(y|x,z) = \mathcal{P}r(y|z)$, where y is the class, x the feature vector, and z the region. Thus no further differentiation can be made within a region. We also assume equal numbers of instances in all regions, making accuracy everywhere equally important.

Fig. 1. Different AL situations, where entropy- or confidence-based uncertainty measures differentiate only on a class' *relative* (vert.) but not on all classes' *total* (horiz.) number of labels.



3 Probabilistic Active Learning

Following the common smoothness assumption [3], we consider that an instance x influences the classification the most in its neighbourhood. Thus, the impact of an additional label primarily depends on the already obtained labels in its neighbourhood. We summarise these by their *absolute* number n, and the share of positives $\hat{p}$ therein, yielding the *label statistics* $\mathit{ls} = (n, \hat{p})$. Here, n is obtained by counting the similar labelled instances for pre-clustered or categorical data (as for the partitions in Figure 1), or approximated by frequency estimates such as kernel frequency estimates for smooth, continuous data. Thus, in x's neighbourhood, n expresses the absolute quantity of labelled information, whereas the density d_x of unlabelled instances quantifies the importance of this neighbourhood, i.e. the share of future classifications that will take place therein compared to other regions of the feature space.

Given a labelling candidate $(x, .)$ from a pool of unlabelled instances $\mathcal{U}$ for a user-specified point classification performance measure [12] like accuracy, we want to compute the expected overall gain in classification performance if requesting its label. This requires knowledge of its label statistics ls, but also of its label y and the true posterior p of the positive class within its neighbourhood. As the latter values of y and p are not directly accessible, we use a probabilistic approach and model Y and P as random variables. This allows us to compute the *expected value* of the gain in performance over all different true posteriors and label realisations, which we denote as *probabilistic gain*[3] (pgain). Finally, we weight it by the neighbourhood's density d_x (over labelled and unlabelled data) to consider the importance of the neighbourhood on the whole data set, quantifying the overall expected performance change. Comparing the overall expected performance change of all candidates, we select the optimal candidate for labelling.

We now first provide the modelling and derive the necessary equations, present the framework of *Probabilistic Active Learning* (*PAL*) with its pseudo-code, and close with discussing its properties.

[3] We do this to differentiate it from the expected gain as in expected error reduction methods like [2], where expectation is solely over label outcomes, but not over the true posterior.

3.1 Probabilistic Gain Calculation

Given a candidate $(x, .)$, the label statistics ls summarise the obtained labels in its neighbourhood. We model the true posterior P of the positive class ($y = 1$) in this neighbourhood as a Beta-distributed random variable, whose realisation p is itself the parameter of the Bernoulli distribution controlling the label realisation $y \in \{0, 1\}$ of any instance within the neighbourhood. Consequently, the number of positives $n \cdot \hat{p}$ among the n already obtained labels in the neighbourhood is the realisation of a Binomial-distributed random variable:

$$P \sim \mathrm{Beta}_{n\cdot\hat{p}+1,n\cdot(1-\hat{p})+1} \tag{1}$$

$$Y \sim \mathrm{Bernoulli}_p = \mathrm{Ber}_p \tag{2}$$

$$(n \cdot \hat{p}) \sim \mathrm{Binomial}_{n,p} \tag{3}$$

The true posterior's Beta distribution above results from its normalised likelihood given the already observed labels, that is

$$\omega_{\mathit{ls}}(p) = \frac{L(p|\mathit{ls})g(p)}{\int_0^1 L(\psi|\mathit{ls})g(\psi)\mathrm{d}\psi} = (1+n) \cdot L(p|\mathit{ls}) \tag{4}$$

$$= \frac{\Gamma(n+2) \cdot p^{n\cdot\hat{p}} \cdot (1-p)^{n\cdot(1-\hat{p})}}{\Gamma(n\cdot\hat{p}+1) \cdot \Gamma(n\cdot(1-\hat{p})+1)} = \mathrm{Beta}_{\alpha,\beta}(p) \tag{5}$$

where the parameters $\alpha = n \cdot \hat{p} + 1$ and $\beta = n \cdot (1 - \hat{p}) + 1$ of the Beta-distribution's pdf $\mathrm{Beta}_{\alpha,\beta}(p)$ are obtained by following a Bayesian approach under a uniform prior for P such that $g()$ is a constant function, and by using the probability mass function according to Eq. 3 for the likelihood $L(p|\mathit{ls})$, and $(1 + n) \cdot \Gamma(n + 1) = \Gamma(n + 2)$.

We take the expectation on the performance gain over these two random variables, yielding the candidate's probabilistic gain (pgain), that defines the expected change of the performance measure for its neighbourhood:

$$\mathrm{pgain}(\mathit{ls}) = \mathtt{E}_p\Big[\mathtt{E}_y\left[\mathrm{gain}_p(\mathit{ls}, y)\right]\Big] \tag{6}$$

$$= \int_0^1 \mathrm{Beta}_{\alpha,\beta}(p) \cdot \sum_{y\in\{0,1\}} \mathrm{Ber}_p(y) \cdot \mathrm{gain}_p(\mathit{ls}, y)\,\mathrm{d}p \tag{7}$$

Here, $\mathrm{gain}_p(\mathit{ls}, y)$ is the candidate's $(x, .)$ performance gain given its label realisation y and the neighbourhood's true posterior p:

$$\mathrm{gain}_p(\mathit{ls}, y) = \mathrm{perf}_p\left(\frac{n\hat{p}+y}{n+1}\right) - \mathrm{perf}_p(\hat{p}) \tag{8}$$

The definition of Eq. 7 and 8 allow the use of any point performance measure (see e.g. [12]) for perf. An example is accuracy (acc), defined as

$$\mathrm{perf}_p(\hat{p}) = 1 - \mathrm{err}_p(\hat{p}) = 1 - \begin{cases} p & \hat{p} < 0.5 \\ 1-p & otherwise \end{cases} \tag{9}$$

where $\mathrm{err}_p(\hat{p})$ is the error rate under Bayes' optimal classification, given a true posterior p and observed posterior $\hat{p}$ of the positive class.

Plugging this in Eq. 7 yields the probabilistic accuracy gain

$$\mathrm{pgain}_{\mathrm{acc}}(\mathcal{ls}) = \\ = \int_0^1 \mathrm{Beta}_{\alpha,\beta}(p) \sum_{y\in\{0,1\}} \mathrm{Ber}_p(y) \left(\mathrm{err}_p(\hat{p}) - \mathrm{err}_p\left(\frac{n\hat{p}+y}{n+1}\right)\right) \mathrm{d}p$$

which we compute by trapezoidal numerical integration over p.

Finally, we weight each candidate's probabilistic gain with the density d_x over *labelled* and *unlabelled* instances in its neighbourhood, and select the candidate with the highest density-weighted probabilistic gain for labelling:

$$x^* = \arg\max_{x\in\mathcal{U}} \left(d_x \cdot \mathrm{pgain}_{\mathrm{acc}}(\mathcal{ls}_x) \right) \tag{10}$$

3.2 PAL Algorithm

The pseudo-code for the resulting probabilistic, pool-based active learning algorithm is given in Figure 2. Iterating over the candidate pool $\mathcal{U}$ (Lines 2-6), for each labelling candidate x one computes its label statistics $\mathcal{ls}_x = (n_x, \hat{p}_x)$, its density weight d_x, and using numerical integration its probabilistic gain, which is weighted by its density weight to obtain g_x. Finally, the candidate with the highest g_x is selected (Line 7).

Fig. 2. The PAL Algorithm

```
1: function POOLBASEDPAL(U,L)
2:     for x ∈ U do
3:         (n_x, p̂_x) ← labelstatistics(x, L)
4:         d_x ← densityweight(x, L ∪ U)
5:         g_x ← pgain((n_x, p̂_x)) · d_x
6:     end for
7:     return arg max_{x∈U}(g_x)
8: end function
```

3.3 PAL's Properties

Statistical Optimality in Disjoint Neighbourhoods For a disjoint neighbourhood concept, like in pre-clustered or categorical data, where instances are partitioned such that instances having an influence on each others' classification belong to the same subset, the density-weighted probabilistic gain of a candidate corresponds precisely to the expected change in overall performance from acquiring the candidate's label. Thus selecting the candidate with highest probabilistic gain is statistically optimal.

For smooth, continuous neighbourhoods, the density-weighted probabilistic gain is the expected change at the candidate's location, serving as an approximation of the overall performance gain. We use this latter concept in our evaluation, as it applies to more data sets and is better comparable to the baseline active learning algorithms.

Computational Efficiency In this subsection, we discuss the asymptotic (with respect to data set size) computational time complexity of PAL and related algorithms for active learning of binary, incremental classifiers. For selecting a candidate from a pool $\mathcal{U}$ of labelling candidates, the PAL algorithm above needs to iterate over all candidates in the pool (Lines 2 – 6). Each iteration consists of 1) querying labelstatstics, 2) querying density weights, and 3) computing the probabilistic gain. The first step requires absolute frequency estimates of labels in the candidate's neighbourhood, similar to the relative frequency estimates needed by entropy or confidence uncertainty measures. These are obtained in constant time by probabilistic classifiers. The second step requires density estimates over all instances, that is over labelled $\mathcal{L}$ and unlabelled $\mathcal{U}$ ones. Precomputing these density estimates once for all later calls of PAL leads to constant query time, as in the pool-based setting the union $\mathcal{L} \cup \mathcal{U}$ is constant. The third step consists of a numeric integration over the true posterior p and a summing over possible label realisations y. Both factors do not depend on the data set size. We used fifty numeric integration steps in all our experiments to get highly precise estimates for expected classification accuracy gain, resulting in a constant factor of $O(50 \cdot 2)$ per probabilistic gain computation. Overall, the iteration over the pool is done in $O(|\mathcal{U}|)$ time.

Selecting the candidate with highest density-weighted probabilistic gain in Line 7 is done in constant time, by using a sweep line approach and storing the maximal value and its corresponding candidate in the previous for-loop.

Overall, PAL requires $O(|\mathcal{U}|)$ time for selecting a candidate from the pool. Uncertainty sampling, using probabilistic classifiers and entropy or confidence uncertainty measures, requires asymptotically the same time, but due the simplicity of its computation with a smaller constant factor involved. In contrast, error reduction as discussed in [15], requires $O(|\mathcal{U}| \cdot |\mathcal{V}|)$ time, where $|\mathcal{V}| \approx |\mathcal{U}|$, as $\mathcal{V}$ needs to be a representative sample of the data.

Characteristics of the Probabilistic Gain For a better understanding of the probabilistic gain function, Figure 3 shows the computed probabilistic gain (in terms of accuracy) for different label statistics, i.e. combinations of different numbers of already obtained labels n and observed posteriors $\hat{Pr}(+|x)$. The following main characteristics of the curve underline its reasonable behaviour:

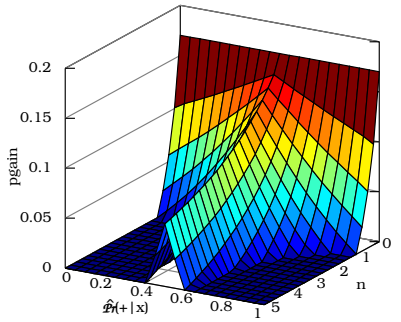
Monotonicity with variable n: With increasing n and a fixed $\hat{Pr}(+|x)$ the probabilistic gain decreases, because it is more likely that the posterior already is correct.

Symmetry with respect to $\hat{Pr}(+|\mathbf{x}) = \mathbf{0.5}$: Evaluating accuracy, pos. and neg. labels count the same, i.e. the probabilistic gain is equal for $\hat{Pr}(+|x)$ and $\hat{Pr}(-|x)$.

Zero for irrelevant candidates: If one label would not change the decision in its neighbourhood, the accuracy remains the same. Thus, gain and probabilistic gain are 0.

This figure is inspired by an illustration of Settles, where different uncertainty measures are plotted as functions of the posterior of a class (see figure 2.4 in [15, p. 15]). Comparing the least confident curve (plot (a) in [15]), it behaves nearly similarly as our probabilistic gain for $n = 1$, but does not change with n.

Fig. 3. Illustration of the probabilistic gain (pgain) as a function of $\hat{Pr}(+|x)$, which is the observed posterior of the positive class, and of n, which is the number of already obtained labels.



4 Experimental Evaluation

From its theoretical characteristics, we expect PAL to be comparable to error-reduction in terms of classification performance, yet faster, and we expect PAL to be better than uncertainty sampling. This section will now verify these characteristics empirically. After outlining the experimental setup, we will discuss the results in the second subsection.

4.1 Evaluation Settings

We compare our new base method PAL with expected error-reduction (in the extended variant proposed by Chapelle in [2], denoted Chap), with uncertainty sampling (using confidence [15] as uncertainty measure, den. Uncer), and with random sampling (den. Rand). While error-reduction is considered as one of the best available AL-methods [15, p. 64], uncertainty sampling is fast and very popular for large or streaming data.

We used Gaussian kernels for frequency estimation, and a Parzen window classifier as in [2] for ensuring comparability with [2]. So, the estimated label frequencies $labelFreq_c$, $c \in \{+, -\}$ at an instance x for the the positive $\mathcal{L}_+$ and the negative class $\mathcal{L}_-$ are calculated by an unnormalised Gaussian function. These frequencies build the label statistics $n = labelFreq_+ + labelFreq_-$ and $\hat{p} = labelFreq_+/n$.

$$labelFreq_c(x) = \sum_{x' \in \mathcal{L}_c} \exp\left(-\frac{\|x' - x\|^2}{2\sigma^2}\right)$$

Our framework starts *without* initial labels, and finishes after 40 label requests. The classifiers, implemented in Octave/MATLAB and run separately on a cluster, use the same pre-tuned, data set-specific bandwidth, and are re-evaluated in each of the 40 steps on the same, dedicated (labelled) test sample. This ensures that only the difference in the active learning strategy is influencing the performance. For better performance assessment, we generated 100 random training and test subsets for each data set, and averaged the results. Evaluation is done on 2 synthetic (based on [2]) and 6 real-world data sets (from [1]). The main characteristics (number of instances, number of attributes), such as training and test set size and the σ of the Parzen window, are summarised in Table 4. The synthetic data sets consist of $4x4$ clusters, arranged in a checker-board formation. While the clusters are low-density-separated in `Che`, they are adjoined in `Che2`. The real-world data sets are Mammographic mass (`Mam`), Vertebral (`Ver`), Haberman's survival (`Hab`), Blood transfusion (`Blo`), Seeds (`See`) and Abalone (`Aba`). All attributes are scaled to a $[0; 1]$-range. We evaluate the performance over the first 40 active label acquisitions and provide the results as learning curves for the optimised performance measure accuracy for all data sets and algorithms.

4.2 Evaluation Results

In accordance to [2] and [15], we provide learning curves in the subfigures of Figure 6. These curves depict the progress in the active classifier's accuracy as 40 training instances are selected one after another for training. This allows to evaluate the performance based on several criteria, and is more informative than tables of the performance at arbitrarily selected learning stages.

(1) When does a curve become flat, i.e. when does the learner converge? On subfigure g) for data set Seeds, the curves become flat already after reading 10 labels, while the curves for data set Checkboard 2 (b) do not converge. Convergence indicates that additional labels do not provide additional use to the classifier, ideally a classifier converges

Dataset	Inst	Attr	$\mathcal{P}r(+)$	|Train|	|Test|	σ
`See`	210	7	33 %	160	50	0.1
`Che`	308	2	44 %	200	108	0.08
`Che2`	392	2	49 %	250	142	0.08
`Hab`	306	3	73 %	256	50	0.1
`Ver`	310	6	32 %	260	50	0.1
`Aba`	4177	8	50 %	400	1177	0.06
`Blo`	748	4	24 %	600	148	0.1
`Mam`	830	11	51 %	630	200	0.1

Fig. 4. Dataset characteristics and parameters (number of instances, number of attributes, proportion of positive instances, training set size, test set size, bandwidth for Parzen window classifier)

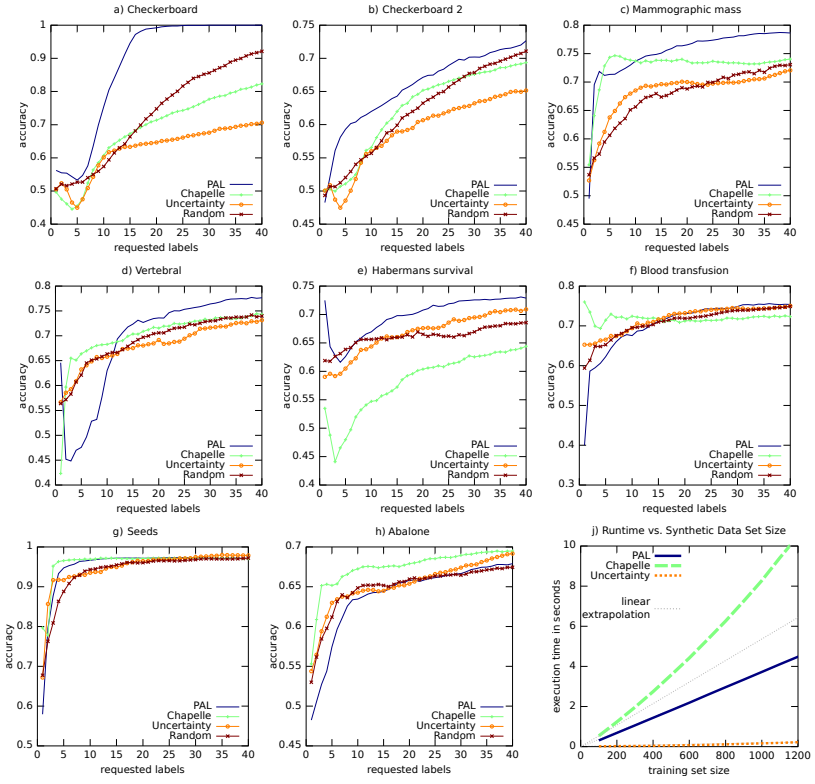
Dataset	PAL	Chap	Uncer	Rand
`See`	0.50	0.93	0.03	0.01
`Che`	0.61	1.16	0.03	0.01
`Che2`	0.92	1.54	0.03	0.02
`Hab`	0.89	1.72	0.03	0.02
`Ver`	0.91	1.84	0.04	0.02
`Aba`	1.51	3.82	0.07	0.04
`Blo`	2.34	6.14	0.1	0.05
`Mam`	2.56	8.48	0.25	0.12

Fig. 5. Average execution time (in seconds), ordering of rows is in ascending training dataset size.

fast and to a high level of performance. This is seen on subfigures a and c, where PAL in contrast to Random Sampling quickly converges to a high performance level.

(2) At what accuracy does a learner stop improving? Clearly, a learner that achieves a 99% accuracy after reading 10 labels is better than one that needs 40 labels to reach the same accuracy value, and also better than one that converges at 75%. Hence, PAL outperforms all other algorithms except on Blood (f), Seeds (g) Abalone (h). The moment of convergence gives also indication on the appropriateness of the data set for active learning. If we contrast subfigures b and g, we must assert that data set Seeds is not truly interesting in terms of active learning: after reading the labels of 5 or at most 10 instances, all learners converge to an accuracy very close to 1. Thus, comparative performance of the active learners on Seeds is not truly informative; this data set is not very appropriate for experiments on active learning (except as a counterexample). The curves on the Blood Transfusion data set (cf. subfigure f) also indicate that active learning is not truly beneficial on this data set.

Fig. 6. a-h: accuracy curves for the algorithms on each dataset; early convergence to very high values is best; improvement after a performance drop is better than a flat curve on low accuracy values; **j**: runtime of PAL on a synthetic data set of varying size (100–1200 candidate instances).



(3) Does a learner recover from previous errors? If a curve becomes flat early, then the learner might be trapped in low accuracy values. This is the case for the algorithm Chapelle on data set Mammographic Mass (c). In contrast, PAL recovers on this data set, as well as on data sets Checkboard, Vertebral and Habermans Survival (a, d, e). Random Sampling never recovers from earlier choices: its performance curves are either flat or go upwards, indicating that an early poor choice cannot be amended. Uncertainty Sampling recovers in some data sets, while Chapelle and PAL always manages to recover if they err in their early choices of label. Summarising the results on accuracy progress, PAL exhibits high performance in all data sets, manages to recover from poor choices and makes best use of available labels, as long as needed (i.e. longer for Checkboard 2 than for Seeds). PAL reaches the best accuracy values on 5 of the data sets, achieves comparable accuracy to the other learners on two data sets (Seeds and Blood Transfusion). PAL is only outperformed once on the Abalone data set.

(4) Execution time The execution time of PAL is shown in Table 5 and plot j of figure 6. Table 5 indicates the execution times of all active learning algorithms on each dataset. We see that PAL achieves better accuracy curves with lower (up to $1/2.5$ times) execution time than the error-reduction algorithm of Chapelle. Nevertheless, the execution time is still significantly higher than that of uncertainty sampling, but like the former its time increases solely linearly with the training set size, i.e. the number of labelling candidates. This is also shown in plot j) of Figure 6, where the execution times on various training set sizes of the same synthetic dataset are plotted. Overall, the uniformly low execution time of uncertainty sampling is accompanied by a stronger variance among the accuracy curves (cf. Figure 6): while PAL has very high performance on all data sets, escapes from earlier errors and exploits well all labels (whenever reasonable, see counterexample on Subfigure 6g), the accuracy curves of Uncertainty Sampling and Random Sampling vary in dependence on the data set. Thus, PAL exhibits stable performance at lower execution time than the expensive error-reduction mechanism, while the simpler algorithms are affected stronger by the idiosyncrasies of the data sets.

5 Conclusion

In this paper, we introduced the probabilistic active learning approach (PAL). It uses probabilistic estimates (label statistics) calculated within the neighbourhood of a labelling candidate. In contrast to Monte-Carlo-based error reduction approach proposed in [13], it models both the true posterior and the candidate's label as random variables. Given a user-specified performance measure, PAL computes the probabilistic gain, that is the expected performance gain over *both* random variables by numeric integration. It subsequently selects the candidate with highest density-weighted probabilistic gain. Like uncertainty sampling [11], PAL requires asymptotically linear time with respect to the pool size, in contrast to quadratic time required by error reduction in [13].

Thus PAL combines two previously incompatible qualities: being fast, and computing and optimising directly a point-performance measure. Given such a user-specified performance measure and the label statistics as input, no additional parameters are required. Our experimental evaluation shows that PAL yields comparable or better classi-

fication performance than error-reduction, uncertainty-sampling or random active learning strategies, while requiring less computational time than error-reduction.

Future work will comprise deriving *specific* closed-form solutions for some point-performance measures such as misclassification loss, as this promises further improvements in speed. Further research is also needed to address *non-myopic* scenarios, where optimising the resulting performance gain from acquiring several labels is required. Finally, as PAL is fast and requires only label statistics but no samples to be kept, its application in *data streams* seems a promising direction for future research.

Acknowledgements

We thank Vincent Lemaire from Orange Labs, France, for the insightful discussions.

References

1. Asuncion, A., Newman, D.J.: UCI ML repository (2013)
2. Chapelle, O.: Active learning for parzen window classifier. In: Proc. 10th Int. Workshop on AI and Statistics. pp. 49–56 (2005)
3. Chapelle, O., Schölkopf, B., Zien, A. (eds.): Semi-Supervised Learning. MIT Press (2006)
4. Cohn, D.: Active learning. In: Sammut, C., Webb, G.I. (eds.) Encyclopedia of ML, pp. 10–14. Springer (2010)
5. Cohn, D.A., Ghahramani, Z., Jordan, M.I.: Active learning with statistical models. J. of AI Research 4, 129–145 (1996)
6. Fu, Y., Zhu, X., Li, B.: A survey on instance selection for active learning. Knowledge and Inf. Syss. 35(2), 249–283 (2012)
7. Garnett, R., Krishnamurthy, Y., Xiong, X., Schneider, J.G., Mann, R.: Bayesian optimal active search and surveying. In: Proc. of the 29th ICML (2012)
8. Gopalkrishnan, V., Steier, D., Lewis, H., Guszcza, J.: Big data, big business: Bridging the gap. In: Workshop on Big Data, Streams and Heterogeneous Source Mining. pp. 7–11 (2012)
9. Ho, S.S., Wechsler, H.: Query by transduction. IEEE Trans. on Pattern A. & Mach. Int. 30(9), 1557–1571 (2008)
10. Krempl, G., Kottke, D., Spiliopoulou, M.: Probabilistic active learning: A short proposition. In: Proc. 21st Europ. Conf. on AI (ECAI2014). IOS Press (2014)
11. Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: Proc. of the 17th ACM SIGIR. pp. 3–12 (1994)
12. Parker, C.: An analysis of performance measures for binary classifiers. In: Proc. of the 11th ICDM. pp. 517 – 526. IEEE (2011)
13. Roy, N., McCallum, A.: Toward optimal active learning through sampling estimation of error reduction. In: Proc. of the 18th ICML. pp. 441–448 (2001)
14. Settles, B.: Active Learning literature survey. CS Tech. Rep. 1648, U. Wisconsin (2009)
15. Settles, B.: Active Learning. No. 18 in Synth. Lect. in AI and ML, Morgan Claypool (2012)
16. Tomanek, K., Morik, K.: Inspecting sample reusability for active learning. In: Guyon, I., et al. (eds.) AISTATS workshop on Act. Learning and Exp. Design. vol. 16, pp. 169–181 (2011)